DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

PhD Preliminary Exam Summary

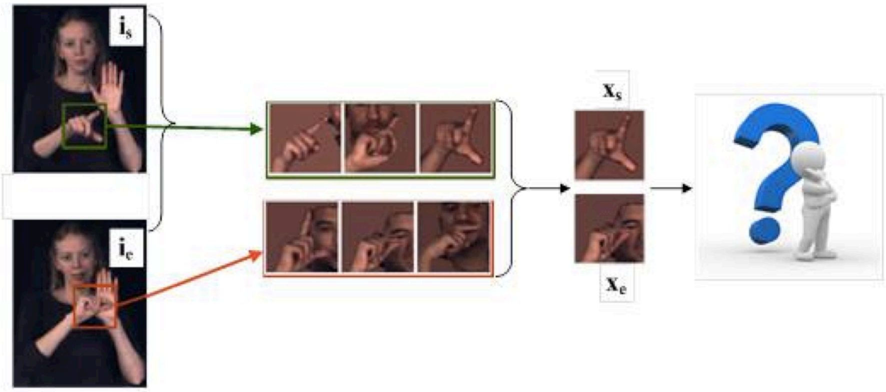
for

American Sign Language (ASL) Recognition

submitted to:

Dr. Joseph Picone, Examining Committee Chair
Dr. Li Bai, Committee Member, Department of Electrical and Computer Engineering
Dr. Seong Kong, Committee Member, Department of Electrical and Computer Engineering
Dr. Rolf Lakaemper, Committee Member, Department of Computer and Information Sciences
Dr. Haibin Ling, Committee Member, Department of Computer and Information Sciences

March 6, 2012



prepared by:

Shuang Lu, PhD Candidate
PhD Advisor: Dr. Joseph Picone, Professor and Chair
Department of Electrical and Computer Engineering
Temple University
College of Engineering
1947 North 12th Street
Philadelphia, Pennsylvania 19122
Tel: 215-204-4841
Email: tuc74165@temple.edu

For further information, please contact Dr. Joseph Picone (email: picone@temple.edu).

EXECUTIVE SUMMARY

Developing sign language applications for hearing-impaired people is extremely important since it is difficult for these people to communicate with people that are unfamiliar with sign language. Ideally, a translation system would improve communication by utilizing common and intuitive signs that can facilitate communications. Continuous sign recognition is significantly challenging since both spatial (hand position) and temporal (when gesture starts/ends) segmentation will cause inaccuracy in the results. Therefore, most research is based on assumptions of knowing either spatial or temporal segmentation, which is not possible for real-time processing.

Frameworks for real-time sign language recognition which do not need precise segmentations are very unstable due in part to a lack of training data. An enhanced level building technique emerged which reduced the requirement for large amounts of training data. This approach reduced the error rate by 54% from 71% to 17%. Unfortunately, error rates increased to above 30% when dealing with complex and unpredictable backgrounds. Also, signer-independent tests, in which no signers are common between the training and test data, resulted in error rates ranging from 21% to 72%. The results suggest that hand shape matching might be a promising approach to improving performance because there are signs with similar positions and motions, but different hand shapes.

The first paper, "*A unified framework for gesture recognition and spatiotemporal gesture segmentation*" by J. Alon, et al., proposes a framework for American Sign Language (ASL) recognition with ambiguous hand position and start/end times. Instead of assuming correct hand positions at each frame, the proposed algorithm searched for a set of several candidate hand locations at each frame, and then hand candidate features were fed into the higher-level model-matching algorithm based on dynamic programming to estimate the hand position. This is referred to as top-down and bottom-up segmentation. Dynamic programming-based approaches, such as Dynamic Time Warping (DTW), have the advantage that only one example is needed, but they lack a statistical model for variations. A hybrid approach was employed in which a Gaussian model for each observation probability was estimated and a uniform transition probability model was used. The Baum-Welch algorithm was used to estimate the parameters of the models for each sign. The proposed approach reduced the false positive rate from 65% to 12% for the sign "Now."

The second paper, "*Handling movement epenthesis and hand segmentation ambiguities in continuous sign language recognition using nested dynamic programming,*" by Yang, Sarkar and Loeding also addresses the same task using a nested DP technique. The framework nests a DP matching algorithm inside an enhanced dynamic level building algorithm. This approach does not need a large training dataset, unlike more sophisticated statistical approaches. Movement epenthesis (meaningless gestures between signs) is also taken into consideration. To reduce the time complexity for DP path searching, a bigram model is used to prune meaningless or unpromising paths. Skin color was modeled using Gaussian Mixture Models (GMMs) and combined with motion cues to find multiple possible hand positions. This resulted in a 40% improvement in performance, reducing the error rate from 82% to 31%.

Both papers used only hand position and motion. Yet, hand shape is also an important feature for distinguishing different signs in ASL. The third paper, "*Exploiting phonological constraints for handshape inference in ASL video,*" Thangali, Nash, Sclaroff and Neidle, proposes a Bayesian network based on a hand shape matching algorithm (HSBN). A novel non-rigid alignment is introduced to reduce the variation caused by slight displacement, rotations and also different implementation habit of signers. A start-end co-occurrence probability is introduced to obtain more possible sign models after acquiring both start and end gesture separately. The *N*-best error rate for the top 5 choices was 38.7% using this approach. The algorithm was planned to be used in conjunction with hand positions and movements to facilitate progress towards person-independent large vocabulary sign recognition.

Table of Contents

1. INTRODUCTION

Developing automated sign language (SL) recognition is important since it is the primary mode of communication for most deaf people. For example, in North America alone it is estimated that as many as 500,000 people use American Sign Language (ASL) as their primary language for communication (Li, et al., 2011). SL recognition also provides an appealing testbed for understanding more general principles governing human motion and gestures. Such gestures are a critical part of a next generation of human-computer interfaces. Moreover, SL is becoming a popular alternative teaching style for babies since they can express feelings by signs much earlier than speaking (Taylor-Dileva, 2010). The development of a system for translating sign language into spoken language would be of great use in a number of applications for the hearing-impaired.

No one form of sign language is universal. Different sign language systems exist throughout the world. For example, unlike the similarities between British English and American English, British Sign Language (BSL) and American Sign Language are two totally different languages and have distinct gestures and rules. However, most sign languages have a similar grammatical structure that enables us to build a generalized SL recognition framework (Sandler & Lillo-Martin, 2001).

SL recognition systems can be classified according to the type of data acquisition employed, the type of recognition task pursued, and the type of features employed, as shown in Figure 1. With respect to data acquisition, there are three main approaches: sensor-based, vision-based and hybrid systems that utilize a combination of sensors and vision systems. Sensor-based SL recognition methods typically use a sensory glove and a motion tracker for detecting hand shapes and body movements (Oz, et al., 2004). Vision-based SL methods use standard cameras, such as those commonly found on many portable computing devices, and rely on image processing and feature extraction techniques for capturing and classifying body movements and hand shapes.

Hybrid systems often integrate data from a range of devices including sensors (often located on a subject's hands), conventional video cameras providing multiple angle views of a subject's hands, and thermo graphic cameras that operate outside the visible light band (e.g., infrared cameras). One popular example of a hybrid system is Microsoft's Kinect sensor (Keskin, et al., 2011) that utilizes a single 2D camera as well as an infrared depth sensor. Kinect can capture color and depth information as part of its measurements.

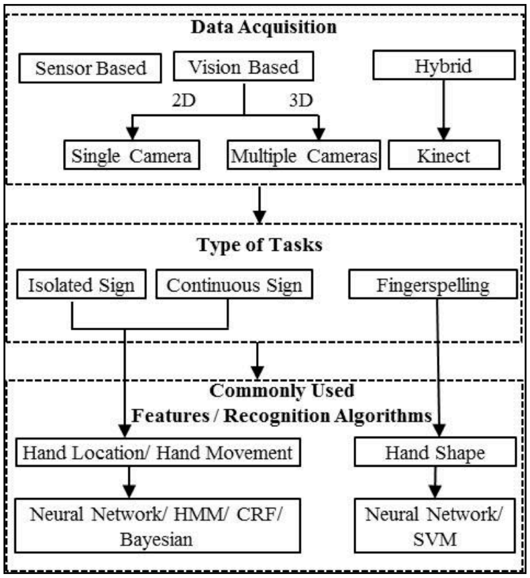


Figure 1. SL recognition tasks are organized by the type of data acquisition, recognition task, feature extraction and pattern recognition algorithm. A new generation of hybrid systems involving the integration of cameras and advanced sensors to measure auxiliary information like depth are emerging.

Sensor-based SL recognition systems have become popular in the last decade as advances in human computer interfaces have fueled a new generation of devices. ASL finger spelling systems were developed using a CyberGlove as a sensor (Sturman & Zeltzer, 1994; Cemil, et al., 2011) and a neural network for feature classification and sign recognition (Kramer, 1996). In 2002, Wan et al. built a Chinese Sign Language (CSL) recognition system based on CyberGloves on both hands. Hidden Markov models (HMMs) of approximately 2,400 phonemes were trained and used to recognize 200 sentences formed by 5119 signs. A word error rate of 7.2% was reported (Gao & Shan, 2002). Mcguire et al. (2004) improved on the neural network approach by using HMMs, achieving a recognition error rate of 6% on an ASL 141-sign vocabulary signed in phrases of four signs using a one-handed glove. In 2007, an ASL recognition system was designed based on linguistic properties with a sensory glove using a neural network, which resulted in a recognition error rate of 8% for a database consists of 60 ASL words (Oz & Leu, 2007).

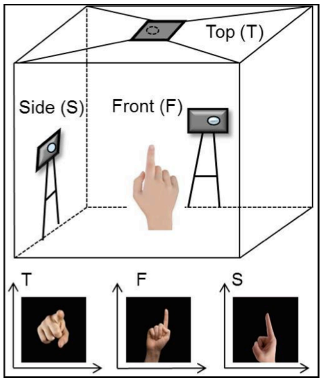


Figure 2. Multiple cameras are used for vision-based gesture recognition to provide hand shape information in three dimensions. Three cameras located in three orthogonal planes are used to reconstruct a 3D image.

Vision-based approaches can be classified into two general categories: a single 2D camera (Ahuja & Tabb, 2002; Ding & Martinez, 2009; Issaacs & Foo, 2004; Athistsos, et al., 2010), and stereo cameras installed at multiple angles (Rodriguez, et al., 1998; Campos & Murray, 2006). Multiple stereo cameras are positioned in three orthogonal planes, as shown in Figure 2, to construct a 3D image. For example, one camera is placed above the hands so that it views the hands looking downward. A second camera is placed in front of the hands. A third camera is placed to the side of the signer. This makes the system very bulky and non-portable. However, the accuracy for both segmentation and recognition improves significantly due to the multiple views.

Recently, Microsoft's Kinect (Keskin, et al., 2011) has been used in hand tracking and gesture classification systems. The Kinect system has enabled a new area of real-time ASL recognition systems (Zafrulla, et al., 2011). Using four-state HMM models and a feature vector that included depth information, a sentence recognition error rate of 65% and a sign recognition error rate of 26% were obtained on a task consisting of 19 signs. Though the Kinect has become extremely popular, there are some issues with the technology. First, the sensor resolution is low, which restricts the position of a signer. If the signer is far away from the sensor, only a few pixels will be assigned to the hands that are insufficient for providing crucial details of finger positions. Second, hand position and orientation have few geometric constraints and are therefore hard to locate with the current generation of the device. Third, a Kinect sensor is much larger than a simple video camera and is also not commonly available as standard equipment on devices such as laptops and phones.

A sensor-based approach is typically more accurate than a vision-based approach since it is much easier to locate finger positions using sensors located on a subject's fingers (Parashar, 2003). However, constraining the user interface through the use of additional sensors often conflicts with the goal of making SL recognition nonintrusive and natural. Hybrid systems attempt to alleviate the need to use specialized sensors on the hands by employing more sophisticated imaging systems. However, these often require a special peripheral (e.g., Kinect), are costly, and not as ubiquitous as standard cameras.

With respect to the task, there are three common tasks reported in the literature: isolated signs, continuous signs and fingerspelling. In an isolated sign task, a subject presents a single sign, typically formed by one or two gestures. The task involves localization of the positions of the hands as well as tracking of their movements. Once the hand locations and movements are identified, the system must select the correct sign from a set of *N* signs using a pattern recognition algorithm (Mcguire, et al., 2004). For an ASL isolated sign task, *N*, the size of the dictionary, is on the order of 6,000 signs.

Continuous signs are sentences or phrases formed by sequencing a series of signs. Therefore, very similar features can be used as isolated signs. However, in the process of transitioning from one sign to the next, the hand shapes and positions for the preceding and following signs are influenced. In other language disciplines, this phenomena is referred to as coarticulation (Cohen & Massaro, 1993). The study of this phenomenon is fairly new to sign language, and the same term is gradually gaining acceptance (Segouat & Braffort, 2010). The general approach to dealing with this problem is to develop context-dependent models of each sign (Vogler & Metaxas, 1997). However, this comes with a great computational cost. Coarticulation is one reason that continuous sign language recognition is very difficult.

An example of isolated and continuous signs is given in Figure 3. Signs for the words "ticket", "buy" and "finish" form a sentence "I have already bought the ticket" which we refer to as a continuous sign. Each individual word is a meaningful sign formed by one or two hand gestures that generally involves movements between gestures. The recognition of continuous signs is harder due to the fact that more gestures and transitions between signs are involved. ASL consists of approximately 6,000 words with unique signs (comparable to morphemes in written language). Additional words are spelled using fingerspelling (Munib, et al., 2007). Similar to written English, ASL has an alphabet of 26 gestures that can be used in fingerspelling. It is very common to use fingerspelling for names, places and specialized terms.

In isolated and continuous sign recognition, handshape features are not typically considered because characterization of hand shapes requires precise segmentation. This is hard to achieve in practice when images have blurred hand movements (hand moves too fast between frames or drift during the process of forming a sign), background scenery that is similar in color to the color of a subject's skin, illumination changes, or moving objects in the background (Yang, et al., 2010). Location and movement features are generally used, which are extracted by hand tracking, motion detection and a variety of segmentation techniques (Bashir, et al., 2005; Alon, et al., 2009).

Fingerspelling, on the other hand, does not need to deal with hand movement. Unlike other SLs, such as British Sign Language, ASL fingerspelling is one-handed, which means only one hand is used when signing the alphabet (Pugeault & Bowden, 2011; Liwicki & Everingham, 2009). This reduces the need for highly accurate hand segmentation due to the fact that both hand positions must be precise for two-handed fingerspellings. The main objective of ASL fingerspelling recognition is to classify alphabet gestures as shown in Figure 4. Therefore, handshape features extracted by edge, corner and pattern detections are often applied (Tanibata, et al., 2002; Hernandez-Rebollar, et al., 2005).

In our work, we will not focus on sensor-based systems because the sensors are still undergoing dramatic changes from a hardware point of view. Our plan is to focus more on machine-learning aspects of the problem. To make the interaction between human and machine simpler and more flexible, we choose to study approaches based on a single 2D camera rather than using multiple cameras. Since hand shape information is important, our first task will be to classify the ASL fingerspelling alphabet. Our work will

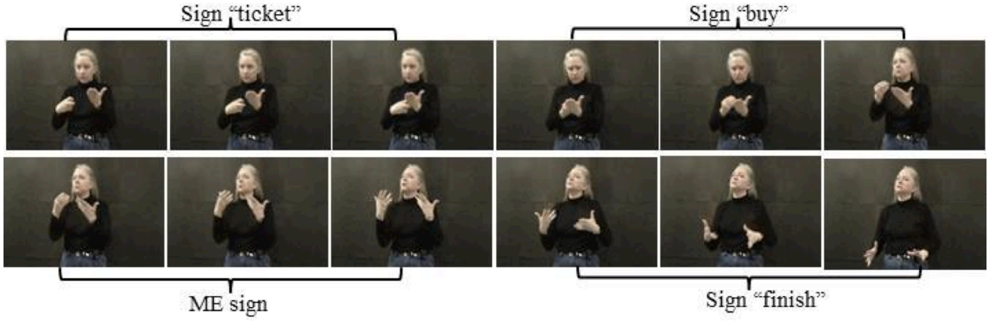


Figure 3. A signer is shown signing the sentence "I have already bought the ticket." This sentence is formed by three signs: "ticket," "buy" and "finish." The three frames between "buy" and "finish" are recognized as movement epenthesis (ME) sign, which refer to movements inserted between two signs that are required to connect them but are not semantically meaningful.

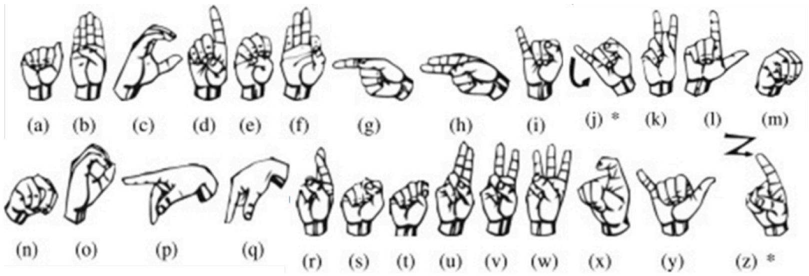


Figure 4. The hand gestures for the 26 signs in the alphabet for ASL. Many of these gestures are very similar (e.g., the gestures for “m” and “n”), making this a very difficult task.

focus on the development of a robust and efficient classification algorithm to distinguish gestures.

An historical summary of ASL recognition approaches and results are shown in Table 1. The earliest work in ASL recognition (Charayaphan & Marble, 1992) was proposed in 1992, which used simple hand tracking techniques and adaptive clustering to classify 31 isolated signs. Neural network-based (NN) approaches were introduced to ASL recognition in the early 1990’s (Wilson, et al., 1993). These used hand location, motion and handshape features as input to an NN for fingerspelling gesture classification. Later, similar work based on NNs combined different data acquisition and feature extraction methods for finger spelling classification (Oz & Leu, 2011). For example, Hamilton, et al. (1994) used a DataGolve with 13 sensors to obtain hand positions. Issacs & Foo (2004) employed wavelet decomposition to extract hand features from 2D images. The best recognition results for NN-based ASL fingerspelling recognition, which used edge detection and a Hough transform for feature extraction, had a classification error rate of 8% for an alphabet of 20 signs (Munib, et al., 2007).

By the mid-1990’s, continuous vision-based sign language recognition based on HMMs became prominent (Starner & Pentland, 1995). Angular cameras were used to generate 3D hand–arm models so that more precise motion and location information could be obtained. Color gloves were employed to improve the accuracy of hand segmentation. Also, a grammar constraint was added between words to decrease the false positive recognition error rate. The error rate for a task involving both a grammar constraint and color gloves was 8% (Starner, et al., 1998). With no gloves or grammar constraints, the error rate increased to 25%.

In 2002, Tanibata, et al. (2002) demonstrated Japanese sign language recognition based on HMMs and obtained a 2% error rate on a task consisting of 65 signs when the face and hands in an image were manually segmented. Yin, et al. (2009) proposed a Segmentally-Boosted HMM (SBHMM) which embedded a discriminative feature selection process into HMM. In SBHMMs, discriminative features that separate the states of HMMs are extracted by a multiclass boosting algorithm. The recognition error rate was reduced to 3.73% from 12.37% on the CyberGlove-based dataset from Mcguire et al. (2004). These experiments indicate that HMM can be applied to SL recognition successfully.

However, most of the algorithms introduced earlier were tested on very small amounts of data. For example, in a study by Munib, et al. (2007) only 10 training and 5 testing images were used for each sign. Error rates for systems that employ hand location and motion features, and use classification algorithms based on HMMs, are less than 20% when tested on 39 signs (Parashar, 2003). However, the error rates increase significantly when the vocabulary size is increased to 147 signs and the segmentations are derived automatically (Yang, et al., 2010). The limited size of the training data is an issue in these studies because the HMMs models for thousands of signs require orders of magnitude more data than is currently available.

Table 1. A summary of related work in ASL recognition is shown. Since the data sets and sensor methodologies vary significantly, it is difficult to directly compare these results. Error rates are still well above 10% for relatively simple signing tasks under realistic operational conditions.

		Vocabulary		
Researchers	**Classification Methods**	**Size (signs)**	**Type**	**Error Rate**
Nguyen et al., 2012	Facial expression, SVM	6 (expression)	Isolated	19.1%
Thangali *et al.,* 2011	Handshape, Bayesian	1500	Isolated	68.9% - 38.7% (Rank 1 – 5)
Pugeault *et al.,* 2011	Kinect, Gabor filter, Random forest	24	FS	47%
Zafrulla *et al.,* 2011	Kinect, PCA, GMM	19	Continuous	24.8% - 48.5%
Yang *et al*., 2010	Level building, ME lable	147	Continuous	17%
Zafrulla *et al.,* 2010	Color gloves, PCA, HMM	19	Continuous	17%
Yin *et al.,* 2009	Sensor gloves, SBHMM	141	Isolated	3.73%
Khambaty *et al.,* 2008	Sensor gloves, Template matching	24	FS	8%
Munib *et al.,* 2007	Hough transform, NN Small size training/test data	20	FS	7.7%
Oz *et al.,* 2007	3D motion tracker, ANN	60	Isolated	5% - 8%
Kong *et al.,* 2007	PCA, HMM	25 (sentences)	Continuous	24% - 33.8%
Yang *et al*., 2006	Key frame extraction, CRF	147	Continuous	19.7%
Mcguire *et al.,* 2004	Sensor gloves, HMM	141	Isolated	6% - 13%
Allen *et al.,* 2003	Sensor gloves, NN, Small size training/test data	24	FS	10%
Parashar, 2003	Motion tracking, PCA, HMM	39	Continuous	5% - 12%
Gupta & Ma, 2001	Geometric features, alignment	10	FS	5.8%
Vogler & Metaxas, 1998	HMM, 3 cameras, data gloves	53	Isolated	8% - 12%
Starner *et al*., 1998	HMM, cameras at angular views, Color gloves, Skin tone	40	Isolated	2% - 8%
Waldron *et al.* , 1995	Neural network	14	Isolated	14%

Real-time continuous sign language recognition using a single 2D camera is a more difficult endeavor compared to many other popular capture devices. Changing illumination, low-quality video, motion blur, low resolution sensors, temporary occlusion, the appearance of a face or other "hand-like" objects, variations in signing behavior and background clutter are all common problems that impede the performance. A framework based on Dynamic Programming (DP) (Alon, et al., 2009) was explored to address those challenges. The task was to retrieve occurrences of ASL signs in a video database consisting of 1,071 signs. Instead of assuming unambiguous and correct hand detection at each frame, the proposed algorithm searched for a set of several candidate hand locations at each frame, and then hand candidate features were fed into the higher-level model-matching algorithm to estimate the hand position.

This is considered a combination of top-down and bottom-up methods (Parashar, 2003). In the bottom-up direction, multiple candidate hand locations are detected and their features are fed into a higher-level model-matching algorithm. In the top-down direction, information from the model is used in the matching algorithm to select, among the exponentially many possible sequences of hand locations, a single optimal sequence. This sequence specifies the hand location at each frame, thus completing the low-level task of hand detection (Alon, et al., 2009). Therefore, the combination of bottom-up and top-down technique generally can improve the accuracy of hand segmentation.

Parameter estimation is problematic in many of these statistical approaches since there is limited training data. Therefore, it is common to assign a priori probabilities for transition probabilities (Alon, et al., 2009) and not to re-estimate these parameters. These approaches resulted in error rates exceeding 50%, especially when movement epenthesis (ME) modeling is taken into consideration. ME modeling, which is used to recognize semantically meaningless frames, can provide better segmentation of each sign within a sentence. In previous work, researchers were trying to model each of ME signs between two different gestures, such as ME for sign AB, AC, etc. However, the possible combinations between gestures are huge, and this results in a combinatorial nightmare for parametric models.

An enhanced level building algorithm (Yang, et al, 2010) which considered ME at each level was introduced for recognizing 147 ASL signs in sentences. The single sign matching process was accomplished by a 2D dynamic time warping (DTW) or 3D dynamic programming matching based on how many hand candidates pair existed in one frame. If there is only one pair of hand candidates found in the image, the algorithm will use 2D DTW to find best match. If multiple pairs are detected within each frame, every possible pair will generate a new path, and the final best match will be the path has least accumulated score. When matching scores between a test hand feature and all sign models are lower than a threshold, the system will assign a ME label to current candidate.

The enhanced level building algorithm reduced error rates by more than 40% when compared to traditional level building and conditional random fields. However, the task described above is based on images collected using simple backgrounds. The error rate increased by at least 10% in experiments which involving complex background scenery (Yang, et al, 2010; Alon, et al., 2009). For example, a dataset with a moving object in the background and a signer wearing short sleeves increased the error rates from 17% to above 30% (Yang, et al., 2010).

Improving the accuracy of single sign matching is crucial since the correctness of each level will affect the overall precision. Most work related to isolated and continuous ASL recognition used only hand position and motion (Bashir, et al, 2005; Wang, et al., 2009; Yang, et al., 2010; Alon, et al., 2009). Yet, hand shape is also an important feature for distinguishing different signs in ASL. Therefore, more recently, researchers are investigating embedding hand shapes into traditional ASL recognition systems (Martines, 2006; Ricco & Tomasi, 2009; Athitsos, et al., 2010). Thangali, et al. (2011) used a histogram of oriented gradient (HOG) features as hand features. Start-end co-occurrence probabilities were computed using a Variational Bayes (VB) network to boost the sign retrieval accuracy.

The error rate for hand shape recognition in this study was relatively high. The correct choice for approximately 80 hand shapes for an isolated sign task did not appear in the top five hypotheses 38.7% of the time for an evaluation dataset of 1500 lexical signs in ASL. The algorithm was planned to be used in conjunction with other articulation parameters (which include hand location, trajectory, and orientation) to facilitate progress towards person-independent large vocabulary sign recognition (Thangali, et al., 2011).

This report is organized in six sections and two appendixes. Section 2 and 3 introduce hand detection and feature extraction techniques. The benefits of applying bottom-up and top-down approaches to sign language recognition is also discussed in section 2. Dynamic programming (DP) based ASL recognition is introduced in section 4. In Section 0, a handshape-based isolated sign recognition system which uses a VB network is discussed. We conclude this report in Section 6 with a discussion of promising future directions. More mathematical details of some of the key algorithms can be found in the appendices.

2. HAND DETECTION

Most existing sign language recognition systems use a hierarchical model that consists of three levels: detection and tracking, feature extraction and recognition (Zaki & Shaheen, 2011; Chen, et al., 2003;

Tanibata, et al., 2002). The detection and tracking layer is responsible for performing temporal data association between successive image frames, so that, at each moment in time, the system knows the locations of the hands. In model-based methods, tracking also provides a way to maintain estimates of model parameters and variables that are not directly observable at a certain moment in time. The feature extraction layer is used for extracting visual features that can be attributed to the presence of hands in the field of view of the cameras. Finally, the recognition layer is responsible for clustering the spatiotemporal data extracted in the previous layers and assigning labels to the resulting clusters representing the associated class of gesture.

Two types of methods have been generally used for hand tracking and detection. One is considered a bottom-up approach (Alon et al., 2009), which uses low-level feature to segment hand regions. This type of algorithm is usually straightforward and not based on prior detection results. However, such approaches are very sensitive to cluttered background and overlap between object. Another type of method is top-down processing (Kumar, Torr & Zisserman, 2010), which is guided by higher level learning processes as the system construct structures based on our experiences and expectations.

2.1. Bottom-up Hand Detection Using Color Models and Motion Tracking

In most dynamic gesture recognition systems, information flows bottom up: the video is input into the analysis module, which estimates the hand pose and shape model parameters, and these parameters are in turn fed into the recognition module, which classifies the gesture. A simple example of bottom-up hand detection process will first extract hand features directly from an input image, and then fit the features into a training and recognition system.

Among all the tasks for gesture and sign language recognition, hand shape and hand motion are the primary sources of information that differentiate one sign from another. Thus, building an efficient and reliable hand detector is the first important step for recognizing signs and gestures (Zhang et al., 2011). Most systems that detect hands from continuous frames place restrictions on the environment (Kolsch & Turk, 2004). For example, a common assumption is that skin color is uniform (Jones & Rehg, 1999). Moreover, many works manually separate hands from other skin-colored objects, especially for cases with insufficient illumination (Binh, Shuichi & Ejima, 2005). Because of the above constraints, hand detection methods based on color cues are not suitable for real world problems.

Motion information is a modality that can mitigate the effects of color distribution and lighting conditions, but this approach becomes increasingly difficult and less reliable for a non-stationary background. Statistical information about hand locations is effective when used as a prior probability, but it requires application-specific training. Shape models generally perform well if there is sufficient contrast between the background and the object, but they have problems especially with non-rigid objects and cluttered backgrounds. In this section, a hand detection approach, which based on both color and motion cues, is introduced.

Since the human skin is relatively uniform, a statistical color model can be employed to compute the probability of every pixel being an acceptable skin color (Zhang, Alonzo & Athitsos, 2011). Jones & Rehg (1999) applied a histogram color model to classify skin and non-skin pixels in images. A database containing 4675 skin colors and 8965 non-skin images were used for training and testing. The skin pixels were manually labeled and then the histogram counts were converted into a discrete probability distribution. A similar histogram was generated for non-skin pixels as well. Both models were then used for maximum likelihood (ML) classification. Motion information is another discriminant cue for hand detection in sign videos since a user needs to move at least one hand to perform a sign. To detect motion, frame differencing was used in which the differences between two consecutive frames was calculated (Gupta & Kulkarni, 2008).

More sophisticated methods, such as optical flow and particle filters (Szeliski, 2011) can be applied instead of frame differencing. However, the computational complexity will increase if more complicated algorithms are used for tracking. A typical system that combines color information with motion cues is shown in Figure 5 (Yang, et al., 2010). A Gaussian Mixture Model (GMM) is used to classify pixels into two clusters that represent skin color and non-skin color. The parameters of the GMM model can be trained using a ML criterion.

Due to the fact that more than one moving object might be detected which has skin-like color, edge detection and other morphology-based pre-processing methods are typically applied to find connected components. For example, a face detection algorithm is first employed to determine the size of the face in an image. Since the sizes of a human face and hand should have some type of relationship, a threshold is then applied to group together candidate pixels within the threshold.

2.2. Hand Detection Using a Combined Bottom-up and Top-down Approach

One common drawback of bottom-up systems is that tracking and recognition typically fail in the absence of perfect hand segmentation (Alon, et al., 2009). However, a top-down approach also has a disadvantage because it emphasizes planning and a complete understanding of the system. Top-down approaches generally use more prior knowledge, typically consisting of domain or application-related constraints, compared to bottom-up approaches.

Therefore, it makes sense to combine bottom-up and top-down process as show in Figure 6. In the bottom-up direction, motion and color cues are used for detecting multiple hand candidates within each frame which as we described in Figure 5. In the top-down direction, information from the model is used in the matching algorithm (HMMs in the example) to select a single optimal sequence among the exponentially many possible sequences of hand locations found from the bottom-up process. After finding an optimal solution, the sequence found will specify the hand location at each frame. The advantage of this combination of bottom-up and top-down approaches is that it reduced the requirement of accurate segmentation, and therefore is more robust to a cluttered background.

3. SIGN FEATURE EXTRACTION

Feature extraction is an essential component of tracking and recognition systems. Selecting good features will result in better accuracy and system performance. Generally, hand shape, hand location, hand movement and 3D hand models are features used for sign language recognition (Rybach, 2006). Three-dimensional hand model-based approaches offer a rich description that allows a wide class of hand gestures. However, a large number of images taken from different views of the hand are required to create a 3D hand model with 27 degree of freedoms (DoFs). Such a model uses five DoFs for the thumb, four for each of the other fingers, and the remaining six DoFs define the global position and rotation of the wrist in the 3D space (Garg, Aggarwal, & Sofat, 2009). Thus, most existing hand feature extraction approaches are focused on 2D features.

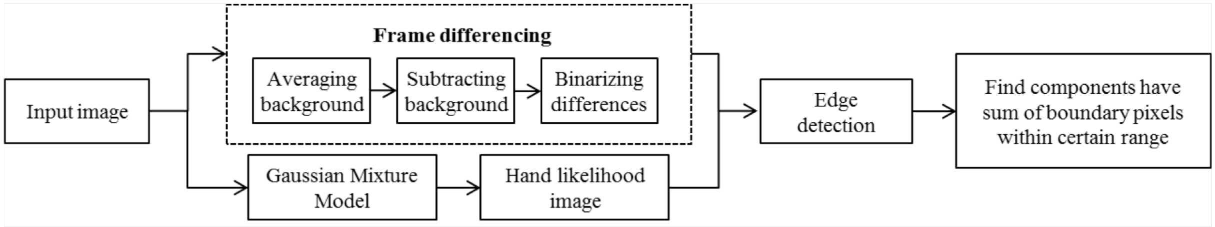


Figure 5. Detection of hand candidates using a GMM classifier and motion information is shown. Edge detection is applied after skin color segmentation

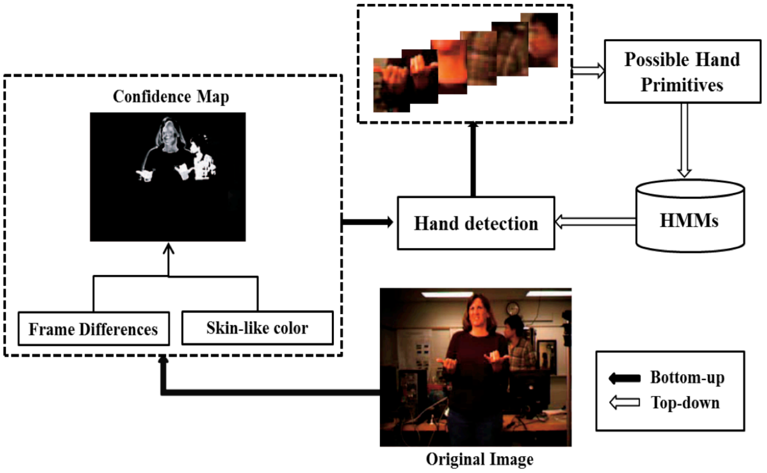


Figure 6. Hand detection method combining bottom-up and top-down approaches. Motion and color information are applied to bottom-up process, and then multiple hand candidates are chosen to be decide later through the top-down step.

3.1. Hand Movement and Location Features

The goal of continuous sign recognition is to translate a sequence of images into meaningful sentences and phrases formed by a series of signs. The features extracted from images can be used for both continuous and isolated sign recognition. Grammar constraints can be employed for continuous sign recognition and can improve the accuracy of hand location detection. For example, if multiple hand candidates have been found from the detection step, grammars can prune meaningless search paths and increase the chance to locate real hands. Hence, isolated sign recognition normally requires more complicated and precise feature extraction algorithms.

Hand positions and velocities are commonly used as primary features in two-dimensional continuous sign language recognition. Many researchers compute local features by using only the center point coordinates of the hand (Yang et al., 2010; Alon et al., 2009). In most cases, the calculation of these features depends on a segmentation of the input image, geometric constraints, and other heuristics. The advantage of the local feature approach is that it only focuses on detected hand region, and therefore is less affected by complex background. However, local methods will fail when the detected region is not accurate, especially when the background image is cluttered and complicated.

In contrast, global features are computed from the whole image, and therefore can provide relationships between the hands and the reference points, such as the position of a head or shoulder, in addition to hand segments (Yang, et al., 2010). Figure 7 shows an example of global hand features proposed by using the center of the face as a reference point. After locating the face and hands in an image, all horizontal and vertical distances between the hand contour points and the center of the face are computed.

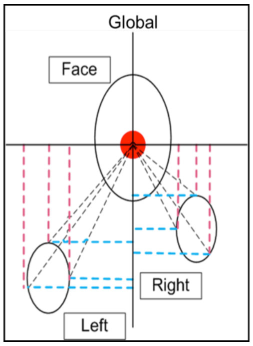


Figure 7. Global feature extraction based on hand positions for dynamic sign recognition. Face detection technique is used to detect face center point as a reference of calculating the distances.

There is a need for a reference point because hand positions can be totally different when the cameras are set up at different angles or positions. In order to calculate distances between candidate hand edge points and a reference point, the hand position of a sign is constrained by the geometric structures of a human body. For example, a one-hand sign with a hand position on the right lower right part of the body will never appear on the left or top side of the face. Hence, the distances between the hands and face should be always within a certain range. One weakness of global feature extraction algorithms is that more non-hand objects may be considered when there is clustered background. Due to the fact that both global and local approaches have drawbacks, more investigations towards feature extraction are needed in the future.

3.2. Handshape Features

ASL consists of approximately 6,000 words with unique signs. Additional words, such as names and places, are spelled using fingerspelling (Munib, et al., 2007). Normally, fingerspelling does not involve any hand movements, which means it is essentially a hand shape recognition problem. In this section, we will introduce one of the most commonly used shape-based feature extraction algorithms – Histogram of Oriented Gradient (HOG) features (Thangali, et al., 2009). These will form the basis for our proposed research.

3.2.1. HOG Feature Extraction

HOG features were first introduced in 2005 for an application involving pedestrian detection (Dalal & Triggs, 2005). In 2009, HOG features were extended to hand gesture recognition as well as many other applications (Wang, el al., 2012; Liwicki & Everingham, 2009). The essential idea behind HOG features is that local object appearance and shape can be described by the distribution of intensity gradients or edge directions.

The first step in calculating HOG features is to compute the gradient intensity, *G*, and orientation, *A*, of each pixel:

$$G_x(x,y) = I(x+1,y) - I(x-1,y) \tag{1}$$

$$G_y(x,y) = I(x,y+1) - I(x,y-1) \tag{2}$$

$$G(x,y) = \sqrt{G_x(x,y)^2 + G_y(x,y)^2} \tag{3}$$

$$A(x,y) = \text{atan}(G_y(x,y)/G_x(x,y)) \ . \tag{4}$$

Next, the entire image is divided into overlapping windows, which are called blocks. Each block consists of four non-overlapped smaller spatial regions named cells. In each cell, *A(x, y)* is quantized in a set of A_r regions by dividing the range [*0*, *2π*] equally. All *G(x, y)* within the same region are summed together to form a 1-D histogram.

Finally, histograms within a block are normalized using the following equation:

$$f_i = \frac{v_i}{\sqrt{\|v\|^2 + 0.01^2}} \ . \tag{5}$$

3.2.2. Hand Image Alignment

For example, if we define the block size to be *90x90* pixels with a 10 pixel overlap, an image with 40×40 pixels will have 64 blocks. Normally, 9 bins are used to calculate the histogram within each cell; however, 12 bins are used in the example from Thangali et al. (2011). Hence, feature vectors from cells in a block are concatenated to form a 48-dimensional HOG feature vector. This vector is then normalized to unit length for robustness to illumination and contrast changes. Thus, the total HOG feature vector will have 64×48 elements in the example shown in Figure 8.

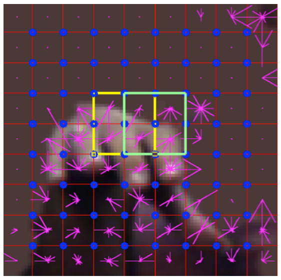


Figure 8. An example of HOG features for a hand gesture is shown. A 50% overlap for each analysis window is typically used.

When matching an observed hand shape image to a labeled hand shape model in the database, similarity scores are used in computing the observation likelihoods. In order to accommodate some of the variations in hand appearance for the same gesture, alignment algorithms can be applied. Thangali et al. (2011) proposed a non-rigid image alignment method. The goal is to find a vector $a^{i \to j}$ (displacement of a point from image *i* to image *j*) that can minimize a total cost, *E*, which consists of two terms, E_{data} and E_{smooth}:

$$a^{i \to j} = \arg\min_a E_{align} = \arg\min_a (E_{data}(a) + E_{smooth}(a)), \qquad (6)$$

where E_{data} is the data association cost and E_{smooth} is the smoothness cost.

The advantage of using a smoothness prior is related to the physical properties of an image: a neighborhood of space or an interval of time are coherence and generally do not change abruptly (Li, 2000). For example, the image in a hand region does not change rapidly over several frames of data. The spatial smoothness prior can be defined as a quadratic function of the predicted displacement vector a, is given by:

$$E_{smooth}(a) = a^T K a, \qquad (7)$$

where $a = [a_1, a_2, ..., a_n]$ and n is the number of control points of an image mesh. Each vector a_n is formed by two elements a_{nx} and a_{ny}, which are the horizontal and vertical displacements of control points *n*. *K* is a stiffness matrix which consists of several local stiffness matrices k_l, which represents the stiffness within each mesh grid. Each sub-matrix k_l is then formed by spring stiffness k_{mn} of spring which connects with end nodes m and n, and is updated in each iteration as:

$$k_{mn} = \frac{k_{base}}{\text{avg}(|a_n| + |a_m|)}, \qquad (8)$$

Where k_{base}, referred to as base stiffness parameter, is typically set experimentally to *75*, *m* and *n* are two end nodes of a spring in the mesh. a_n and a_m are the positions of m and n. More details, including an algorithm implementation, can be found in Thangali et al. (2011).

By combining equation (6) and (7), we get:

$$\mathrm{E}_{\mathrm{align}}(\mathrm{a}) = \mathrm{E}_{\mathrm{data}}(\mathrm{a}) + \mathrm{a}^{\mathrm{T}}\mathrm{Ka}. \tag{9}$$

The cost function reaches its optimal when:

$$\nabla_{\mathrm{a}}\,\mathrm{E}_{\mathrm{align}}(\mathrm{a}) = 0 \Rightarrow -\nabla_{\mathrm{a}}\,\mathrm{E}_{\mathrm{data}}(\mathrm{a}) = \mathrm{Ka}. \tag{10}$$

Using the gradient descent (Yuan, 2008) method:

$$\mathrm{a}^{\mathrm{i}} = \mathrm{a}^{\mathrm{j}} \text{-} \alpha\nabla\,\mathrm{E}_{\mathrm{data}}(\mathrm{a}). \tag{11}$$

Let f_{a} be the local displacements to decrease E_{data}:

$$f_{\mathrm{a}} = \mathrm{a}^{\mathrm{i}\rightarrow\mathrm{j}} = \mathrm{a}^{\mathrm{i}} \text{-} \mathrm{a}^{\mathrm{j}} = -\alpha\nabla\,\mathrm{E}_{\mathrm{data}}(\mathrm{a}). \tag{12}$$

Combining equations (10) and (12), we have:

$$f_{\mathrm{a}} = \alpha\mathrm{Ka}. \tag{13}$$

An overview of this algorithm is shown in Figure 9. The position vectors a_n^i and a_m^i of two control points m and n in image i corresponds to a_n^j and a_m^j in image j. First, the initial displacement vectors $\mathrm{a}^{init:i\rightarrow j}$ are calculated. A search window W is defined which is centered at each control point of image i. Within the search window, HOG features are calculated by sliding two pixels vertically or horizontally each time as shown in Figure 9(c). A Euclidean distance is used to compute E_{data} at each point.

After calculating E_{data} at all points within a search window, one point is randomly selected from points that have 5 lowest scores for E_{data}. The position of this point is then assigned as the initial new position for the control point in the new image, which is initial value for $\mathrm{a}_{\mathrm{n}}^{init:i\rightarrow j}$. One advantage of the random selection is that it reduces the chance of falling into local minimum. With displacement vectors $\mathrm{a}^{init:i\rightarrow j}$, equation (8) and (13), we can obtain the value for $\alpha\mathrm{a}$. Finally, a line search is applied to decide the value of α to minimize E_{data}(a), which will also provide the final result for vector $\mathrm{a}^{i\rightarrow j}$.

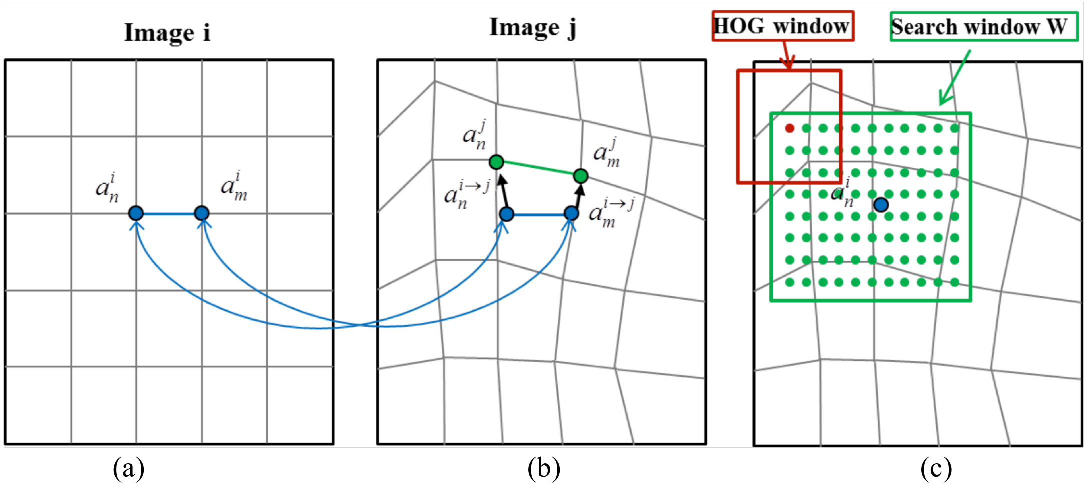


Figure 9. The non-rigid image alignment process with smoothness prior adaptation: (a) shows the undeformed mesh and control points; (b) shows the new positions of corresponding control points from image *i*, and the displacement vectors; and (c) shows the places for calculating E_{data} within the search window, *W*.

4. CONTINUOUS ASL RECOGNITION BASED ON DP

Dynamic programming (DP) (Silverman & Morgan, 1990) has been an important sequential-decision analysis tool for speech recognition systems since the 1960's. It is also widely used to solve a variety of computer vision problems, such as, stereo matching, hand writing recognition and gesture recognition (Alon et al., 2009). DP is a general approach for solving problems exhibiting two properties: optimal substructure and overlapping sub-problems (Cormen et al., 2001). Optimal substructure means that optimal solutions of sub-problems can be used to find the optimal solutions of the overall problem.

In ASL recognition, the goal of matching a sentence of signs to a query subsequence is to find several candidate hand sequences that can be best mapped to several model sequences. The main idea of DP-based continuous ASL recognition is that the main problem can be broken down into sub-problems of computing matching costs between each hand image sequence and a hand model. The matching costs computed for these sub-problems can then be combined to compute the optimal matching cost for the entire sentence. One advantage of DP-based algorithms is that they can handle sequences of different lengths; time alignment and time warping are included in the optimization process. For example, two image sequences with five and ten frames each can be recognized using the same model.

4.1. Dynamic Time Warping and Hidden Markov Models

Dynamic Time Warping (DTW) and Hidden Markov Models (HMM) are two well-known non-linear sequence alignment or pattern matching algorithms (Fang, 2009). DTW is used to compute a distance between two time series. Standard DTW is based on the idea of deterministic DP. However, more real-world signals are stochastic processes, such as speech, video, etc. Hence, a new algorithm called "stochastic DTW" was proposed in 1988. In this method, conditional probabilities are used instead of local distances in standard DTW, and transition probabilities instead of path costs. This actually is very similar to an HMM model.

An HMM is a statistical model in which the system being modeled is assumed to be a Markov process with unknown parameters (Rabiner, 1989). The challenge is to determine the hidden parameters from the observable data. The extracted model parameters can then be used to perform further analysis included pattern recognition applications. An HMM can be considered as the simplest dynamic Bayesian network. In a regular Markov model, a state is directly visible to the observer, and therefore the state transition probabilities are the only parameters that need to be estimated. In a hidden Markov model, the state is not directly visible, but variables influenced by the state are visible (Fang, 2009).

For an unknown input ASL sign with N image frames, every path from the start state to the exit state of the HMM which passes through exactly N emitting HMM states is a potential recognition hypothesis. Each of these paths has a log probability which is computed by summing the log probability of each individual transition in the path and the log probability of each emitting state generating the corresponding observation. Within-HMM transitions are determined from the HMM parameters, while between-model transitions are determined by the language model likelihoods. The objective is to find the path through the network that has the highest log probability. The Baum-Welch (Rabiner, 1989) algorithm, a special case of the Expectation-Maximization (EM) approach, is usually used for HMM parameter estimation. Details of the Baum-Welch algorithm can be found in Welch (2003).

Template-based approaches like DTW have an advantage that only one example is needed, but lack a statistical model for variations. On the other hand, higher accuracy is expected when using more expressive dynamic models, such as HMMs. However, these models require a large amount of training data to learn their parameters (Alon et al., 2009). Though it is possible to estimate state output probabilities of HMMs using a process similar to what was used in DTW systems, learning state transition probabilities and language model likelihoods from small amount of training data is not possible. Therefore, Alon et al. (2009) proposed a hybrid approach, which estimated a Gaussian model for the observation probabilities (like an HMM), but employed the uniform transition probability model of DTW. This method can be considered as a simplified stochastic DTW, and can be implemented as follows:

Suppose $I = (I_1, I_2, ..., I_j)$ is a query sequence from a test video. At each frame j, we can extract K feature vectors $\{Q_{j1}, Q_{j2},..., Q_{jk}\}$. Each vector includes a 2D hand position and a 2D hand velocity. Let's also assume we have gesture models $X = (X_1, X_2, ..., X_g)$, and each gesture model has m states. For each state X_i^g, a Gaussian observation density (μ_i^g, Σ_i^g) which assigns a likelihood to the observation vector Q_{jk} is obtained by the Baum-Welch algorithm. Here, μ_i^g, Σ_i^g are the mean and covariance matrix of the feature vectors observed in state X_i^g. The mission for matching video with a model is to calculate a cost function $d(i,j,k) = d(X_i^g, Q_{jk})$ which is a Mahalanobis distance:

$$d(i,j,k) = (Q_{jk} - \mu_i^g)^{'}(\Sigma^g)^{-1}(Q_{jk} - \mu_i^g). \tag{14}$$

DTW is used to map each image frame to a state of a hand model, so the total sum of distances of the query sequence is minimized. This algorithm is useful for a task with a small training dataset; however, more complicated stochastic models should be applied to achieve better performance when more data is available.

As mentioned in Sections 2 and 3, the features normally used for continuous signs matching are hand locations and velocities. If multiple hand candidates are found in one image, we need to record the matching path of all hand candidates at each frame. This changes the 2D DTW algorithm into a 3D dynamic programing process. The only difference between 2D DTW and 3D DP is that 3D process needs to compare more alternatives at each step.

4.2. An Improved Pruning Method for DP

One issue with the above 3D dynamic programming matching approach is that the time complexity will increase dramatically when more gesture models and states of the model are applied. For example, if N_j hand candidates are found from frame *j*, then the number of possible hand pairs (representing the left and right hand) will be $N_{pair} = N_j(N_j - 1) = N_j^2 - N_j$. The higher N_{pair} is, the more complicated the recognition process will become, because more potential paths will be added to the computation. Thus, eliminating improbably or unlikely paths is an essential way to maintain computational efficiency.

The process of removing low-scoring partial paths from the search space is known as pruning. A number of heuristic criteria can be applied to identify such paths and to set the appropriate thresholds on path scores which keep only qualified paths for future steps. Some commonly used heuristics are: beam search, limiting the total number of model instances active at a given frame and setting an upper bound on the number of models allowed to end at a given frame (Deshmukh, Ganapathiraju, & Picone, 1999). The most commonly used method is beam search.

In beam search, a predetermined likelihood value, referred to as beam width, is chosen at each frame, and all paths with a matching score larger than the beam width are removed from further consideration. However, the value of beam width at each step is not easy to define. One possible way of doing this is by calculating distances between a model state and training feature vectors that are matched with the model state, and set the beam width to be the maximum distance (Alon et al., 2009).

If the maximum matching distance at *cell*(i, j, k) from the training data is τ_i and the test distance $d(i, j, k)$

at this cell is larger than τ_i, all paths that pass through *cell*(*i, j, k*) will be eliminated (pruned). When lacking large amount of training data, many nodes in the test data may have a larger value than the beam widths in the training data. This could potentially prune too aggressively and delete the optimal path. To avoid this, Alon et al. (2009) defined a parameter ε derived from cross-validation training and added it to each τ_i, so the final threshold for each cell should be $\tau_i^{'} = \tau_i + \varepsilon$. This cross validation approach reduces the chance of over pruning and also decreases the computational complexity of the search process.

4.3. Enhanced Level Building for ASL Recognition

DP-based algorithms have been widely used to solve various kinds of optimization problems. Two crucial problems in video-based sign language and gesture recognition systems can be solved by dynamic programming. The first problem occurs at the highest level (e.g., sentence). Movement epenthesis (ME) (Yang, et al., 2010), which means the necessary but meaningless movement between signs, can result in difficulties in modeling and scalability as the number of signs increases. In the past, ME gestures had only been modeled explicitly such that each ME between two signs was trained as a specific sign. This creates a major problem because millions of ME signs need to be learned when the vocabulary size is large. The second problem occurs at the loweest level (e.g., feature). Ambiguity of hand detection and occlusion will propagate errors to higher levels. Regarding the above issues, Yang et al. (2010) constructed an enhanced level building (eLB) framework that can handle both of these problems based on a DP approach.

The classic Level Building algorithm refers to a search process that is performed at various levels, where a level corresponds to the positions of the gesture units within the possible sentence. At each level, we maximize the score over all unit models for every frame *t* and find a best hypothesis. The search at the next level starts with the winning score of the previous level. After going through all levels, all hypothesis sequences found at the end frame of the query will be compared to each other and the optimum solution which has the best score will be selected as the result.

The eLB algorithm proposed by Yang et al. (2010) used the classic Level Building algorithm with a threshold set to decide whether there is an ME gesture. At each frame, if the highest matching score of a test sequence with all meaningful gestures is less than a threshold, an ME label is going to be added instead of a modeled gesture. This raises a question of how to calculate the cost for an ME label and threshold. The author defined the cost as follows:

$$D(S_{v+k}, T(j+1, m)) = (m-j)\alpha, \tag{15}$$

where α is a penalty that decides the threshold for a good match, j+1 and *m* are the start and the end frame of a new level, *S* is corresponding to a certain sign model. The variable k represents the length of the ME label, which means S_{v+k} represents an ME sign with 2 frames.

A general function for scoring at each level is:

$$A(l,i,m) = \begin{cases} D(S_i, T(1:m)), & if\ l=1, \\ \infty, & \forall i\ s.t.\ R(p,i)=0, \\ \min\limits_{k,j} A(l-1,k,j) + D(S_i, T(j+1:m)), & otherwise, \end{cases} \tag{16}$$

where *D* is the matching cost between a single sign and a segment of the test sequence, and $R(i, j)$ represents the local constraint:

$$R(i,j)=\begin{cases} 1, & \text{if } S_i \text{ can be the predecessor of } S_j \\ 0, & \text{if } S_i \text{ cannot be the predeccessor of } S_j. \end{cases} \tag{17}$$

This local constraint is similar to N-gram (Deshmukh, Ganapathiraju, & Picone, 1999) in speech recognition with N equal to 2.

After the optimal path is obtained, backtracking is applied to reconstruct the optimal sign sequence. An array ψ is used to store the best matched sign at each level, which is defined as

$$\psi(l,i,m)=\begin{cases} -1, & \text{if } l=1, \\ -1, & \forall i \text{ s.t. } R(p,i)=0, \\ \arg\min A(l-1,k,j)+D(S_i,T(j+1:m)), & \text{otherwise.} \end{cases} \tag{18}$$

Suppose we have in total 100 frames for a test sequence. The eLB implementation steps are shown in Figure 10:

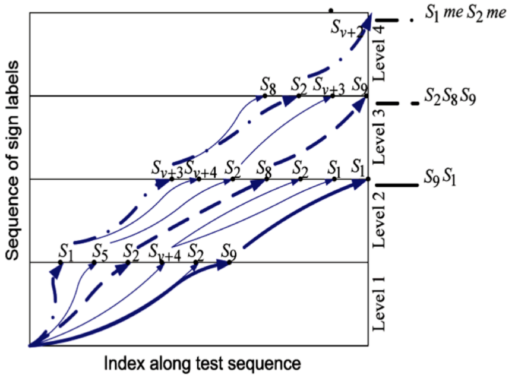

Sequence of sign labels
Index along test sequence
Level 1
Level 2
Level 3
Level 4
S_1 S_5 S_2 S_{v+4} S_2 S_9
S_{v+3} S_{v+4} S_2 S_8 S_2 S_1 S_1
S_8 S_2 S_{v+3} S_9
S_{v+2}
S_1 me S_2 me
$S_2 S_8 S_9$
$S_9 S_1$

Figure 10. One example of the enhanced level building matching . S1, ME, S2, ME is finally decided after comparing with S2, S8, S9 and S9, S1 sequences due to lowest total cost

Level 1:

$$A(1,i1,j1)=D(S_{i1},T(1:j1)) \tag{19}$$

Possible Sign Number (*i1*)	1	5	2	V+4	2	9
Possible sign end frame (*j1*)	40	55	65	80	85	90

By minimizing $A(1,i1,j1)$ at each possible end frame, we would find several possible signs for the first level.

$$\min(A(1,i1,j1))=\begin{cases} D(1,T(1:10)), & j1=10, \\ D(5,T(1:20)), & j1=20, \\ D(2,T(1:30)), & j1=30, \\ D(V+4,T(1:50)), & j1=50, \\ D(2,T(1:60)), & j1=60, \\ D(9,T(1:70)), & j1=70. \end{cases} \tag{20}$$

Level 2:

$$\begin{aligned} A(2,i2,j2) &= \min A(1,i1,j1) + D(S_{i2},T(j1+1:j2)) \\ &= \min D(S_{i1},T(1:j1)) + D(S_{i2},T(j1+1:j2)) \end{aligned} \tag{21}$$

Possible Sign Number (*i2*)	V+3	V+4	2	8	2	1	1
Possible sign end frame (*j2*)	40	55	65	80	85	90	100

By minimizing $A(2,i2,100)$, we would find possible signs for the second level

$$\min(A(2,i2,j2)) = \begin{cases} D(1,T(1:10)) + D(V+3,T(11:40)), & j2 = 40, \\ D(1,T(1:10)) + D(V+4,T(11:55)), & j2 = 55, \\ D(5,T(1:20)) + D(2,T(21:65)), & j2 = 65, \\ D(2,T(1:30)) + D(8,T(31:80)), & j2 = 80, \\ D(V+4,T(1:50)) + D(2,T(51:85)), & j2 = 85, \\ D(2,T(1:60)) + D(1,T(51:90)), & j2 = 90, \\ D(9,T(1:70)) + D(1,T(71:100)), & j2 = 100. \end{cases} \tag{22}$$

Level 3:

$$\begin{aligned} A(3,i3,j3) &= \min A(2,i2,j2) + D(S_{i3},T(j2+1:j3)) \\ &= \min[\min(D(S_{i1},T(1:j1) + D(S_{i2},T(j1+1:j2)))] + D(S_{i3},T(j2+1:j3)) \end{aligned} \tag{23}$$

Possible Sign Number (*i3*)	8	2	V+3	9
Possible sign end frame (*j3*)	65	80	90	100

$$\min(A(3,i3,j3)) = \begin{cases} D(1,T(1:10)) + D(V+3,T(11:40)) + D(8,T(41:65)), & j3 = 65, \\ D(1,T(1:10)) + D(V+3,T(11:40)) + D(2,T(41:80)), & j3 = 80, \\ D(5,T(1:20)) + D(2,T(21:65)) + D(V+3,T(66:90)), & j3 = 90, \\ D(2,T(1:30)) + D(8,T(31:80)) + D(9,T(81:100)), & j3 = 100. \end{cases} \tag{24}$$

Level 4:

$$\begin{aligned} A(4,i4,j4) &= \min(A(3,i3,j3) + D(S_{i4},T(j3+1:j4))) \\ &= \min\{\min[\min(D(S_{i1},T(1:j1)) + D(S_{i2},T(j1+1:j2)) \\ &\quad + D(S_{i3},T(j2+1:j3)) + D(S_{i4},T(j3+1:j4)))]\} \end{aligned} \tag{25}$$

Possible Sign Number (*i4*)	V+2
Possible sign end frame (*j4*)	100

$$\begin{aligned} \min(A(4,i4,j4)) &= D(1,T(1:10)) + D(V+3,T(11:40)) \\ &\quad + D(2,T(41:80)) + D(V+2,T(81:100)), \qquad j4 = 100. \end{aligned} \tag{26}$$

As we can see from the example, the best match, which the traditional LB algorithm would find, is {S2, S8, S9}, whereas the real sign sequence should be {S1, S2}. By applying the eLB algorithm with ME signs, the recognized sequence is {S1, ME, S2, ME}, which matches the original sign exactly.

Dynamic programming-based approaches, like DTW, have the advantage that only one example is needed, but they lack a statistical model for variations. On the other hand, higher accuracy is expected when using more expressive dynamic models, such as HMM or conditional random field (CRF). When process a sign sentence, accurate allocation of ME gestures has proved to enhance the recognition results by Yang et al. (2010). They also found that sign features with only hand locations and motions limit the discriminative abilities of the recognition system. Hence, richer features in conjunction with hand shape and facial expression may provide better performance.

5. HANDSHAPE INFERENCE FOR SIGN MATCHING

As mentioned by Yang et al. (2010), sign recognition methods based on only hand positions and movements are not robust because hand shape is an important component of sign language recognition. Thus, recent research has been focusing on how to use hand shape information to develop sign or hand gesture recognition (Oz, et al., 2011; Keskin, et al., 2011; Khambaty, etl al., 2008). In speech recognition, a language model, which models the co-occurrence probabilities of several words in a sentence, is usually used to enhance the recognition accuracy. Similar to the language model, the probabilities of two gestures being start and end gestures of an isolated sign also follow a certain distribution. Hence, Thangali et al. (2011) proposed a Variational Bayesian (VB) network which models the co-occurrence of start and end gesture pairs to improve the recognition accuracy.

5.1. Handshape Bayesian Network (HSBN)

An overview of the approach in the paper is shown in Figure 12. Given an input test hand pair $\{i_s, i_e\}$, we want to match it with a corresponding model hand pair $\{x_s, x_e\}$. This can be seen as maximizing the likelihood, $P(x_s, x_e \mid i_s, i_e)$:

$$\begin{aligned} P(x_s, x_e \mid i_s, i_e) &= \frac{1}{P(i_s, i_e)} P(x_s, x_e, i_s, i_e) \\ &= \frac{1}{P(i_s, i_e)} P(i_s \mid x_s) P(i_e \mid x_e) P(x_s, x_e) \\ &\propto P(x_s \mid i_s) P(x_e \mid i_e) \frac{P(x_s, x_e)}{P(x_s) P(x_e)}. \end{aligned} \tag{27}$$

In the above equation, $P(x_s \mid i_s)$ and $P(x_e \mid i_e)$ are calculated using:

$$P(x_s \mid i_s) \overset{define}{\propto} \sum_{i=1}^{k} e^{-\beta i} \delta(x_{DB}^i, x_s), \tag{28}$$

where, k is the number of examples retrieved from a database by a k-nearest neighbor algorithm, β is a decaying weight, and δ is an indicator function which tests whether the end frame of x_{DB}^i is x_s. $P(x_s)$ and $P(x_e)$ are the marginal probability of $P(x_s, x_e)$. Therefore, the problem becomes how to find the value of $P(x_s, x_e)$.

An important and difficult problem in Bayesian inference is computing the marginal probability. The marginal probability is an important quantity because it allows us to select between several model structures. It is a difficult quantity to compute because it involves integrating over all parameters and latent variables, which usually results in a complex integral in a high dimensional space. Most simple approximations have failed catastrophically at this (Beal & Ghahramani, 2003).

5.2. Variational Bayesian Learning in an HSBN

Variational methods have recently become popular in the context of inference problems. Variational Bayes is a particular variational method (Jordan, et al., 1999) which aims to find some approximate joint

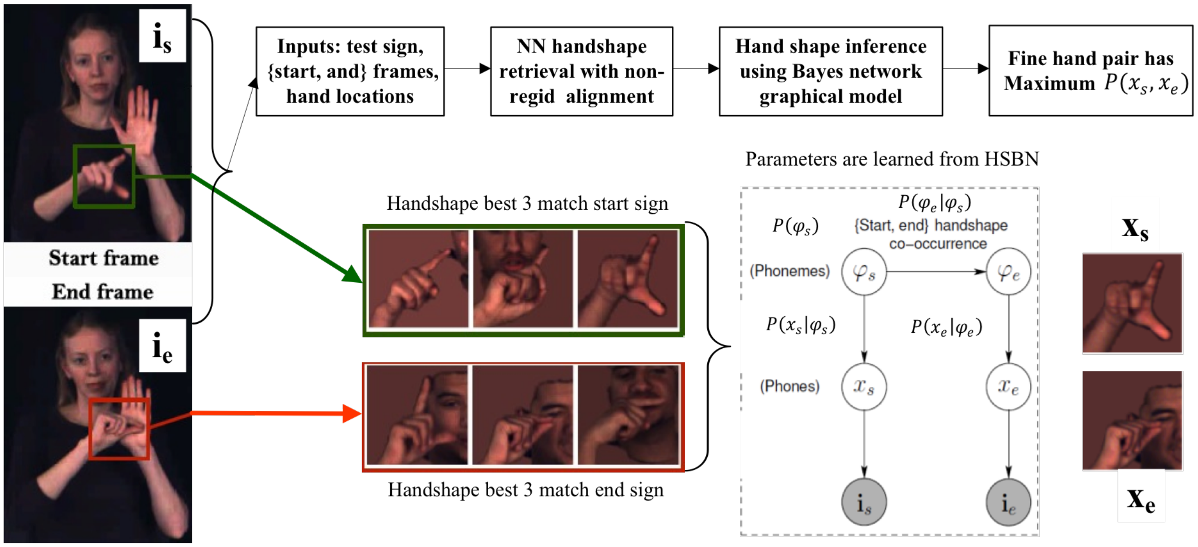

i_s
Start frame
End frame
i_e
Inputs: test sign, {start, and} frames, hand locations
NN handshape retrieval with non-regid alignment
Hand shape inference using Bayes network graphical model
Fine hand pair has Maximum $P(x_s, x_e)$
Parameters are learned from HSBN
Handshape best 3 match start sign
Handshape best 3 match end sign
$P(\varphi_s)$
$P(\varphi_e|\varphi_s)$
{Start, end} handshape co-occurrence
(Phonemes)
φ_s
φ_e
$P(x_s|\varphi_s)$
$P(x_e|\varphi_e)$
(Phones)
x_s
x_e
i_s
i_e
$\mathbf{x_s}$
$\mathbf{x_e}$

Figure 12. An illustration of the whole proposed HSBN approach. Best three match gestures for start and end signs are found by matching process including non-rigid alignment process, and then VB inference are applied for retrieving the sign with most probable start-end gesture pair.

distribution $Q(x,\theta)$ over hidden variables x to approximate the true joint distribution $P(x)$, and defines 'closeness' as the KL divergence $KL[Q(x,\theta)||P(x)]$ (Fox & Roberts, 2011). It maximizes the likelihood by iteratively increasing a lower bound. For example, the marginal likelihood $P(x_s,x_e)$ in equation (27) can be calculated as:

$$P(x_s, x_e) = \sum_{\varphi_s,\varphi_e} \pi_{\varphi_s} \mathrm{a}_{\varphi_s,\varphi_e} b^s_{\varphi_s}(x_s) b^e_{\varphi_e}(x_e). \tag{29}$$

The parameters $\lambda = \{\pi, \mathrm{a}, \mathrm{b_s}, \mathrm{b}_e\}$ above correspond to the following multinomial probability distributions:

$$\begin{aligned} &\pi_{\varphi_s} = P(\varphi_s);\ \mathrm{a}_{\varphi_s,\varphi_e} = P(\varphi_e \mid \varphi_s); \\ &b^s_{\varphi_s}(x_s) = P(x_s \mid \varphi_s);\ b^e_{\varphi_e}(x_e) = P(x_e \mid \varphi_e), \end{aligned} \tag{30}$$

where $\{\varphi_s, \varphi_e\}$ are the {start, end} hand shape categories which are considered as hidden states in the network, and $\{x_s, x_e\}$ are the observed hand shape pairs which contains different realizations of $\{\varphi_s, \varphi_e\}$. Thus, the hidden variable φ_i corresponds to x_i, which includes all possible implementations of a sign model in the HSBN, as shown in Figure 11. The advantage of using a hidden layer for this task is that it can adapt to the variations of hand shapes caused by the signing habit of different signers. It may also be less sensitive to hand rotations to other existing algorithms. To approximate the marginal probability distribution, the EM algorithm is employed to maximize the lower bound. The goal of the EM algorithm (Dempster, et al., 1977) is to estimate the model parameter(s) for which the observed data are most likely. Each iteration of the EM algorithm

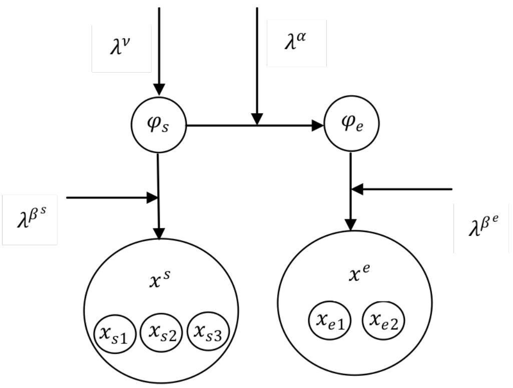


Figure 11. One-to-many associations between hidden and observed variables for HSBN. Any start or end parameter can correspond to more than one observation.

consists of two processes: the E-step and the M-step. In the expectation, or E-step, the missing data are estimated given the observed data and current estimate of the model parameters. This is achieved using the conditional expectation. In the M-step, the likelihood function is maximized under the assumption that the missing data are known. The estimate of the missing data from the E-step is used instead of the actual missing data. Convergence is assured since the algorithm is guaranteed to increase the likelihood at each iteration (Borman, 2004).

To maximize the likelihood function, the VB-EM approach employs a lower bound function $\mathcal{F}$ which is derived as follows:

$$\ln P(x) = \ln \int d\lambda P(x \mid \lambda) P(\lambda) \tag{31}$$

$$= \ln \int d\lambda Q_{\lambda}(\lambda) P(x \mid \lambda) \frac{P(\lambda)}{Q_{\lambda}(\lambda)} \tag{32}$$

$$\geq \int d\lambda Q_{\lambda}(\lambda) \ln P(x \mid \lambda) \frac{P(\lambda)}{Q_{\lambda}(\lambda)} \tag{33}$$

$$= \int d\lambda Q_{\lambda}(\lambda) [\sum_{i=1}^{N} \ln P(x_i \mid \lambda) + \ln \frac{P(\lambda)}{Q_{\lambda}(\lambda)}] \tag{34}$$

$$= \int d\lambda Q_{\lambda}(\lambda) [\sum_{i} \ln \sum_{\varphi_i} P(x_i, \varphi_i \mid \lambda) + \ln \frac{P(\lambda)}{Q_{\lambda}(\lambda)}] \tag{35}$$

$$\geq \int d\lambda Q_{\lambda}(\lambda) [\sum_{i} \sum_{\varphi_i} Q_{\varphi_i}(\varphi_i) \ln \frac{P(x_i, \varphi_i \mid \lambda)}{Q_{\varphi_i}(\varphi_i)} + \ln \frac{P(\lambda)}{Q_{\lambda}(\lambda)}] \tag{36}$$

$$= \mathrm{F}\ (Q_{\lambda}(\lambda), Q_{\varphi_i}(\varphi_i)). \tag{37}$$

The derivation from equation (32) to (33) and equation (35) to (36) is based on Jensen's inequality (Dempster, et al., 1977).

Jensen's inequality states that a convex function of the variable expectation is larger than or equal to the expectation of the convex function of the same variable. We know that log function is a concave function (Carter, 2001), so we have:

$$\ln E[x] \geq E[\ln(x)]. \tag{38}$$

By simply taking functional derivatives with respect to each of the Q(·) distributions and equating these to zero, we get the distributions that maximize $\mathcal{F}$. Synchronous updating of the variational posteriors is not guaranteed to increase $\mathcal{F}$ but consecutive updating of dependent distributions is. The result is that each update is guaranteed to monotonically and maximally increase $\mathcal{F}$. Taking the derivative of Lower bound function $\mathcal{F}$ from equation (36) with the respect of $Q_{\lambda}(\lambda)$ and $Q_{\varphi}(\varphi)$, we have

$$\begin{aligned} &\frac{\partial \mathrm{F}}{\partial Q_{\lambda}(\lambda)} = 0 \Rightarrow \\ &\qquad \ln Q_{\lambda}(\lambda) = \sum_{i} \sum_{\varphi_i} Q_{\varphi_i}(\varphi_i) [\ln P(x_i, \varphi_i \mid \lambda) - \ln Q_{\varphi_i}(\varphi_i)] + \ln P(\lambda) - C_{Q_{\lambda}}; \end{aligned} \tag{39}$$

$$\begin{aligned} &\frac{\partial \mathrm{F}}{\partial Q_{\varphi_i}(\varphi_i)} = 0 \Rightarrow \\ &\qquad \ln Q_{\varphi_i}(\varphi_i) = \int d\lambda Q_{\lambda}(\lambda) \ln P(x_i, \varphi_i \mid \lambda) - C_{Q_{\varphi_i}}. \end{aligned} \tag{40}$$

C_{Q_λ} and $C_{Q_{\varphi_i}}$ here are normalizing constants for the variational distributions.

The complete data log-likelihood can be expanded given the model in Figure 11:

$$\ln P(x_i, \varphi_i \mid \lambda) = \ln \pi^i_{\varphi_s} + \ln a_{\varphi^i_s \varphi^i_e} + \sum_{j=1}^{|x_i|} \ln b^s_{\varphi^i_s}(x^{ij}_s) + \sum_{j=1}^{|x_i|} \ln b^e_{\varphi^i_e}(x^{ij}_e). \tag{41}$$

The prior distributions for model parameters are chosen from the Dirichlet family. The Dirichlet distribution (Appendix A.2) is one that has often been utilized in Bayesian statistical inference as a convenient prior distribution. The most common reason for using a Dirichlet distribution is that it is from the same family as multinomial distribution (Huang, 2005), and they are a conjugate prior. If the data has multinomial distribution and the prior of the parameters of the data is a Dirichlet distribution, then the posterior distribution of the data parameters is also Dirichlet. The benefits of this are that the posterior distribution is easy to compute and updating parameters normally does not involve complicated integration.

Based on the properties of Dirichlet distribution (Beal, 2003), we have:

$$\ln P(\lambda) = \ln \mathrm{Dir}(\{\pi, a, b^s, b^e\} \mid \{v^o, a^o, \beta^{so}, \beta^{eo}\}) \tag{42}$$

$$= \ln \mathrm{Dir}(\pi \mid v^o) + \sum_{\varphi_s} \ln \mathrm{Dir}(a_{\varphi_s} \mid a^o_{\varphi_s}) + \sum_{\varphi_s} \ln \mathrm{Dir}(b^s_{\varphi_s} \mid \beta^{so}_{\varphi_s}) + \sum_{\varphi_s} \ln \mathrm{Dir}(b^s_{\varphi_s} \mid \beta^{so}_{\varphi_s}) \tag{43}$$

$$\begin{aligned} &= \sum_{\varphi_s} (v^o_{\varphi_s} - 1) \ln \pi_{\varphi_s} + \sum_{\varphi_s, \varphi_e} (a^o_{\varphi_s, \varphi_e} - 1) \ln a_{\varphi_s, \varphi_e} \\ &\quad + \sum_{\varphi_s, x} (\beta^{so}_{\varphi_s}(x) - 1) \ln b^s_{\varphi_s}(x) + \sum_{\varphi_e, x} (\beta^{eo}_{\varphi_e}(x) - 1) \ln b^e_{\varphi_e}(x). \end{aligned} \tag{44}$$

Substituting equations (41) and (44) into equation (39), we get:

$$\begin{aligned} \ln Q_\lambda(\lambda) = &\sum_i \sum_{\varphi^i_s, \varphi^i_e} Q_{\varphi^i_s, \varphi^i_e}(\varphi^i_s, \varphi^i_e)[\ln \pi_{\varphi^i_s} + \ln \mathrm{a}_{\varphi^i_s, \varphi^i_e} + \sum_{j=1}^{|x_i|} \ln b^s_{\varphi^i_s}(x^{ij}_s) + \sum_{j=1}^{|x_i|} \ln b^e_{\varphi^i_e}(x^{ij}_e)] \\ &- \sum_i \sum_{\varphi_i} Q_{\varphi_i}(\varphi_i) \ln Q_{\varphi_i}(\varphi_i) + \ln P(\lambda) - 1 \end{aligned} \tag{45}$$

$$\begin{aligned} = &\sum_i \sum_{\varphi^i_s} Q_{\varphi^i_s}(\varphi^i_s) \ln \pi^i_{\varphi_s} + \sum_i \sum_{\varphi^i_s, \varphi^i_e} Q_{\varphi^i_s}(\varphi^i_s, \varphi^i_e) \ln a_{\varphi^i_s, \varphi^i_e} \\ &+ \sum_i \sum_{\varphi^i_s} Q_{\varphi^i_s}(\varphi^i_s) \sum_{j=1}^{|x_i|} \ln b^s_{\varphi^i_s}(x^{ij}_s) + \sum_i \sum_{\varphi^i_e} Q_{\varphi^i_e}(\varphi^i_e) \sum_{j=1}^{|x_i|} \ln b^e_{\varphi^i_e}(x^{ij}_e) \\ &+ \sum_{\varphi_s} (v^o_{\varphi_s} - 1) \ln \pi_{\varphi_s} + \sum_{\varphi_s, \varphi_e} (a^o_{\varphi_s, \varphi_e} - 1) \ln a_{\varphi_s, \varphi_e} \\ &+ \sum_{\varphi_s, x} (\beta^{so}_{\varphi_s}(x) - 1) \ln b^s_{\varphi_s}(x) + \sum_{\varphi_e, x} (\beta^{eo}_{\varphi_e}(x) - 1) \ln b^e_{\varphi_e}(x) + C_{Q_\lambda} \end{aligned} \tag{46}$$

$$
\begin{aligned}
&= \sum_{\varphi_s} (v^o_{\varphi_s} + \sum_i Q^i_{\varphi_s}(\varphi_s) - 1) \ln \pi_{\varphi_s} \\
&\quad + \sum_{\varphi_s, \varphi_e} (a^o_{\varphi_s, \varphi_e} + \sum_i Q_{\varphi^i_s, \varphi^i_e}(\varphi_s, \varphi_e) - 1) \ln a_{\varphi_s, \varphi_e} \\
&\quad + \sum_{\varphi_s, x} (\beta^{so}_{\varphi_s}(x) + \sum_i \sum_{j=1}^{|x_i|} \delta(x, x^{ij}_s) Q^i_{\varphi_s}(\varphi_s) - 1) \ln b^s_{\varphi_s}(x) \qquad (47) \\
&\quad + \sum_{\varphi_e, x} (\beta^{eo}_{\varphi_e}(x) + \sum_i \sum_{j=1}^{|x_i|} \delta(x, x^{ij}_e) Q^i_{\varphi_e}(\varphi_e) - 1) \ln b^e_{\varphi_e}(x) \\
&= \ln \mathrm{Dir}(\pi \mid v^*) + \sum_{\varphi_s} \ln \mathrm{Dir}(a_{\varphi_s} \mid a^*_{\varphi_s}) + \sum_{\varphi_s} \ln \mathrm{Dir}(b^{so}_{\varphi_s} \mid \beta^{s*}_{\varphi_s}) + \sum_{\varphi_e} \ln \mathrm{Dir}(b^{eo}_{\varphi_e} \mid \beta^{e*}_{\varphi_e}), \qquad (48)
\end{aligned}
$$

where,

$$
\begin{aligned}
v^* &= \sum_{\varphi_s} v^*_{\varphi_s} = \sum_{\varphi_s} [v^o_{\varphi_s} + \sum_i Q_{\varphi^i_s, \varphi^i_e}(\varphi_s, \varphi_e)] \\
a^*_{\varphi_s} &= \sum_{\varphi_e} a^*_{\varphi_s, \varphi_e} = \sum_{\varphi_e} [a^o_{\varphi_s, \varphi_e} + \sum_i Q_{\varphi^i_s, \varphi^i_e}(\varphi_s, \varphi_e)] \\
\beta^{s*}_{\varphi_s} &= \sum_x \beta^{s*}_{\varphi_s}(x) = \sum_x [\beta^{so}_{\varphi_s} + \sum_i \sum_{j=1}^{|x_i|} \delta(x, x^{ij}_s) Q_{\varphi^i_s}(\varphi_s)] \, . \\
\beta^{e*}_{\varphi_e} &= \sum_x \beta^{e*}_{\varphi_e}(x) = \beta^{eo}_{\varphi_e} + \sum_i \sum_{j=1}^{|x_i|} \delta(x, x^{ij}_e) Q_{\varphi^i_e}(\varphi_e)
\end{aligned}
$$

Using what we obtained above, $Q_\lambda(\lambda)$ can be decomposed as the sum of Dirichlet distributions. Therefore, equation (40) is equal to:

$$
\begin{aligned}
\ln Q_{\varphi_i}(\varphi_i) = -C_{Q_{\varphi_i}} &+ \int d\pi \, \mathrm{Dir}(\pi \mid v^*) \ln \pi_{\varphi^i_s} + \int da_{\varphi^i_s} \, \mathrm{Dir}(a_{\varphi^i_s} \mid a^*_{\varphi^i_s}) \ln a_{\varphi^i_s, \varphi^i_e} \\
&+ \sum_{j=1}^{|x_i|} \int db^s_{\varphi^i_s} \, \mathrm{Dir}(b^s_{\varphi^i_s} \mid \beta^{s*}_{\varphi^i_s}) \ln b^s_{\varphi^i_s}(x^{ij}) + \sum_{j=1}^{|x_i|} \int db^e_{\varphi^i_e} \mid \beta^{e*}_{\varphi^i_e} \ln b^e_{\varphi^i_e}(x^{ij}).
\end{aligned} \qquad (49)
$$

Using the identity $\int d\pi \, \mathrm{Dir}(\pi \mid v) \ln \pi_i = \psi(v_i) - \psi(\sum_k v_k)$, ($\psi$ is digamma function, see Appendix A.5),

$$
\begin{aligned}
\ln Q_{\varphi_i}(\varphi_i) = -C_{Q_{\varphi_i}} &+ \psi(v^*_{\varphi^i_s}) - \psi(\sum_k v^*_k) + \psi(a^*_{\varphi^i_s, \varphi^i_e}) - \psi(\sum_k a^*_{\varphi^i_s, k}) \\
&+ \sum_{j=1}^{|x_i|} [\psi(\beta^{s*}_{\varphi^i_s}(x^{ij}_s)) - \psi(\sum_k \beta^*_{\varphi^i_s, k})] + \sum_{j=1}^{|x_i|} [\psi(\beta^{e*}_{\varphi^i_e}(x^{ij}_e)) - \psi(\sum_k \beta^*_{\varphi^i_e, k})].
\end{aligned} \qquad (50)
$$

Now, we go back to equation (36), and apply equation (40) to it. Then we obtain:

$$
\mathrm{F}\ (Q_\lambda, Q_{\varphi_i}) = \sum_i C_{Q_{\varphi_i}} - \int d\lambda Q_\lambda(\lambda) \ln \frac{Q_\lambda(\lambda)}{P(\lambda)} \qquad (51)
$$

$$\begin{aligned} &= \sum_i C_{Q_{\varphi_i}} - KL(v^* \,\|\, v^o) - \sum_{\varphi_s} KL(a_{\varphi_s}^* \,\|\, a_{\varphi_s}^o) \\ &\quad - \sum_{\varphi_s} KL(\beta_{\varphi_s}^{s*} \,\|\, \beta_{\varphi_s}^{so}) - \sum_{\varphi_e} KL(\beta_{\varphi_e}^{e*} \,\|\, \beta_{\varphi_e}^{eo}), \end{aligned} \tag{52}$$

where $KL(\cdot \,||\cdot)$ is K-L convergence function (Appendix A.3).

The EM algorithm will repeat the above steps iteratively until changes in the value of F $(Q_\lambda, Q_{\varphi_i})$ are below a threshold. With the lower bound F $(Q_\lambda, Q_{\varphi_i})$ learned by the variational approach mentioned above, the probability of {start, end} co-occurrence can then be obtained. One major contribution of this proposed HSBN algorithm is that it takes into consideration both {start, end} hand shape co-occurrence probabilities which increases recognition performance similar to the way a language model influences performance in speech recognition (Picone, 1990).

6. CONCLUSIONS AND FUTURE WORK

This report summarized and compared state of the art ASL recognition systems from three aspects: hand detection, feature extraction, and gesture recognition. Accurate hand detection generally requires precise segmentation of an image. However, this is hard to achieve when the background is complicated and skin color varies. Almost all existing ASL recognition systems or demos tend to constrain the background to be plain. Still, it is impossible to always limit the background conditions in real world applications. Therefore, Alon et al. (2009) and Yang et al. (2010) applied a combination of bottom-up and top-down approaches which allowed multiple hand position hypotheses within each image frame. With the stochastic modeling ability of top-down algorithms, the final detected hand locations are more robust in cluttered backgrounds.

In the past, hand positions and movements were frequently used for continuous and isolated sign recognition, while hand shape was more meaningful for fingerspelling. However, many continuous gestures have the same hand locations and movements but different hand shapes, and therefore can only be differentiated by the shapes of the hands. Hence, more research interests have focused on the feature extraction of hand shapes.

Histogram of Oriented Gradient, as one of the most popular shape representation algorithms, has been successfully applied to hand gesture recognition. It uses distributions of gradients to reflect the edge information, which does not reply on pre-segmentation and is more robust to illumination changes. Despite all the benefits HOG has, it is not scale and rotation invariant and is sensitive to backgrounds containing subjects with clear edges. Handshape-based recognition still needs further investigation, which should be one of the major developments of ASL recognition in the following decades.

The dynamic programming-based gesture recognition system has been very popular because it is flexible to match sign sequences with different lengths. DTW, as one of the most commonly used DP-based algorithms, has many similarities with HMM. Ideally, as the data collected of real-world signs are stochastic signals, HMM should over perform DTW for ASL recognition application. However, DTW is more generally used, due to the fact that there is generally not enough data available for training parameters needed for stochastic models. Thus, finding a dataset with a greater amount of data for testing HMM-based systems is part of future work.

Continuous ASL recognition often involves movement epenthesis between two meaningful signs, which is hard to model when a database has a large vocabulary. Yang et al (2010) embedded the recognition of ME signs into a level building algorithm which avoided the process of modeling it. Though algorithms with multiple levels may obtain better accuracy compared to one level DTW, the computation needed for the whole system also increases. As a result, multiple constraints should be considered to either speed up the training and recognition process or improve accuracy when using dynamic programming approaches.

Similar to speech recognition, ASL recognition can be separated into several levels: state level, hand shape level, and sign level. At each level, certain types of pruning algorithms, such as, beam search, can be applied to reduce the computational complexity and the recognition error rate. Modeling the linguistic constraints on the co-occurrence of hand shapes in lexical signs can also improve the robustness of the recognition systems.

As the hand is a non-rigid object, there are variations in the production of a hand shape articulated by the same or different signers. Because of this, a set of hidden variables is normally introduced to the modeling process. After adding the hidden variables into the computation process, it is often difficult to calculate the likelihood probabilities using integrals. Variational Bayesian methods provide an alternative way of computing probabilities, which can be generalized to other algorithms, including HMM.

In conclusion, without sophisticated sensors, vision-based ASL recognition is a very challenging research topic. A better hand feature representation will be the first task in order to develop a reliable ASL recognition system. Advanced statistical modeling algorithms (instead of simple DTW) need to be investigated to improve the recognition process, which means datasets with larger amounts of samples are required. Finally, more research on reducing the effects caused by hand shape and background variation is necessary.

7. REFERENCES

Alon, J., Athitsos, V., Yuan, Q., & Sclaroff, S. (2009). A Unified Framework for Gesture Recognition and Spatiotemporal Gesture Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(9), 1685-1699.

Athitsos, V., Wang, H., & Stefan, A. (2010). A Database-based Framework for Gesture Recognition. *Personal and Ubiquitous Computing*, 14(6), 511-526.

Bashir, F., Qu, W., Khokhar, A., & Schonfeld, D. (2005). HMM-based Motion Recognition System Using Segmented PCA. *Proceedings of the International Conference on Image Processing* (pp. 1288-1291).

Beal, M. J. (2003). *Variational Algorithms for Approximate Bayesian Inference (Doctoral dissertation*, University College London). Retrieved from http://www.cse.buffalo.edu/faculty/mbeal/thesis/.

Beal, M. J., & Ghahramani, Z. (2003). The Variational Bayesian EM Algorithm for Incomplete Data: with Application to Scoring Graphical Model Structures. *Bayesian Statistics 7* (pp. 453–464). Oxford University Press.

Binh, N. D., Shuichi, E., & Ejima, T. (2005). Real-Time Hand Tracking and Gesture Recognition System. *Proceedings of International Conference on Graphics, Vision and Image Processing* (pp. 362–268). Louisville, Kentucky, USA.

Borman, S. (2004). *The Expectation Maximization Algorithm. A Short Tutorial.*

Campos, T. de & Murray, D. (2006). Regression-based Hand Pose Estimation from Multiple Cameras. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 782-789).

Carter M. (2001). *Concave and convex functions*. Retrieved from: http://michaelcarteronline.com/FOME/pdf/ConcaveFunctions.pdf

Charayaphan, C., & Marble, A. (1992). Image Processing System for Interpreting Motion in American Sign Language. *Journal of Biomedical Engineering*, 14(5), 419-425.

Chen, F., Fu, C., & Huang, C. (2003). Hand Gesture Recognition Using a Real-Time Tracking Method and Hidden Markov Models. *Image and Vision Computing*, 21(8), 745–758.

Cohen, M. M., & Massaro, D. W. (1993). Modeling Coarticulation in Synthetic Visual Speech. *Models and Techniques in Computer Animation* (pp. 139-156). Springer-Verlag.

Cormen, T. H., Stein, C., Rivest, R. L., & Leiserson, C. E. (2001). *Introduction to Algorithms*. McGraw-Hill.

Dalal, N., & Triggs, B. (2005). Histograms of Oriented Gradients for Human Detection. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (Vol. 1, pp. 886–893).

Deshmukh, N., Ganapathiraju, A., & Picone, J. (1999). Hierarchical search for large-vocabulary conversational speech recognition: working toward a solution to the decoding problem. *IEEE Signal Processing Magazine*, 16(5), 84–107.

Ding, L., & Martinez, A. (2009). Modelling and Recognition of the Linguistic Components in American Sign Language. *Image and Vision Computing*, 27(12), 1826-1844.

Fang, C. (2009). *From Dynamic Time Warping (DTW) to Hidden Markov Model(HMM)* (pp. 1–7).

Farhadi, A., & Forsyth, D. (2006). Aligning ASL for Statistical Translation Using a Discriminative Word Model. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (Vol. 2, pp. 1471-1476).

Felzenszwalb, P. F., & Zabih, R. (2011). Dynamic Programming and Graph Algorithms in Computer Vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(4), 721-740.

Feng, Z., Yang, B., Chen, Y., Zheng, Y., Xu, T., Li, Y., Xu, T., et al. (2011). Features Extraction from Hand Images Based on New Detection Operators. *Pattern Recognition*, 44(5), 1089–1105.

Fox, C., & Roberts, S. (2011). A Tutorial on Variational Bayesian Inference. *Artificial Intelligent Review*, 38(2), 1-13.

Gao, W., & Shan, S. (2002). An Approach Based on Phonemes to Large Vocabulary Chinese Sign Language Recognition. *Proceedings of Fifth IEEE International Conference on Automatic Face Gesture Recognition* (pp. 411-416).

Garg, P., Aggarwal, N., & Sofat, S. (2009). Vision Based Hand Gesture Recognition. *World Academy of Science, Engineering and Technology*, 49, 972–977.

Gupta, L. (2001). Gesture-based Interaction and Communication: Automated Classification of Hand Gesture Contours. *IEEE Transactions on Systems, Man and Cybernetics, Part C (Applications and Reviews)*, 31(1), 114-120.

Gupta, K., & Kulkarni, A. V. (2008). Implementation of An Automated Single Camera Object Tracking System Using Frame Differencing and Dynamic Template Matching. *In T. Sobh (Ed.), Advances in Computer and Information Sciences and Engineering* (pp. 245–250).

Hamilton, J., & Micheli-Tzanakou, E. (1994). Alopex Neural Networks for Manual Alphabet Recognition. *Proceedings of the International Conference of the IEEE Engineering in Medicine and Biology Society* (pp. 1109-1110).

Hamsici, O. C., & Martinez, A. M. (2009). Active Appearance Models with Rotation Invariant Kernels. *Proceedings of IEEE International Conference on Computer Vision* (pp. 1003-1009).

Hernandez-Rebollar, J. (2005). Gesture-driven American Sign Language Phraselator. *Proceedings of the International Conference on Multimodal Interfaces* (Vol. 1, pp. 288-292).

Huang, J. (2005). *Maximum Likelihood Estimation of Dirichlet Distribution Parameters. Distribution*. CMU Technique Report, 1- 9.

Huenerfauth, M. (2006). Representing Coordination and Non-coordination in American Sign Language Animations. *Behaviour & Information Technology*, 25(4), 285-295.

Huenerfauth, M., & Lu, P. (2010). Accurate and Accessible Motion-Capture Glove Calibration for Sign Language Data Collection. *ACM Transactions on Accessible Computing*, 3(1), 1-32.

Isaacs, J., & Foo, S. (2004). Hand Pose Estimation for American Sign Language Recognition. *Proceedings of the Southeastern Symposium on System Theory* (pp. 132-136).

Jones, M. J., & Rehg, J. M. (1999). Statistical Color Models with Application to Skin Detection. *Proceedings of 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (pp. 274–280).

Jordan, M. I., Ghahramani, Z., Jaakkola, T. S., & Saul, L. K. (1999). An Introduction to Variational Methods for Graphical Models. *Machine Learning*, 37, 182–233.

Keskin, C., Kirac, F., Kara, Y. E., & Akarun, L. (2011). Real Time Hand Pose Estimation Using Depth Sensors. *Proceedings of the International Conference on Computer Vision* (pp. 1228-1234).

Khambaty, Y., Quintana, R., Shadaram, M., Nehal, S., Virk, M. A., Ahmed, W., & Ahmedani, G. (2008). Cost Effective Portable System for Sign Language Gesture Recognition. *Proceedings of the IEEE International Conference on System of Systems Engineering* (pp. 1-6).

Kolsch, M., & Turk, M. (2004). Robust Hand Detection. *Proceedings of the Sixth IEEE International Conference on Automatic Face and Gesture Recognition* (pp. 614–619).

Kumar, M. P., Torr, P. H. S., & Zisserman, A. (2010). OBJCUT: Efficient Segmentation Using Top-Down and Bottom-Up Cues. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(3), 530–545.

Li, K., Lothrop, K., Gill, E., & Lau, S. (2011). A Web-Based Sign Language Translator Using 3D Video Processing. Proceedings of the International Conference on Network-Based Information Systems (pp. 356-361).Liwichi, S., & Everingham, M. (2009). *Automatic Recognition of Fingerspelled Words in British Recognition Workshops* (pp. 50-57).

Liwicki, S., & Everingham, M. (2009). Automatic Recognition of Fingerspelled Words in British Sign Language. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (pp. 50–57).

Li, S. Z. (2000). *Modeling Image Analysis Problems Using Markov Random Fields*. Elsevier Science, 20, 1–43.

M., G., Menon, R., Jayan, S., James, R., & G.V. V., J. (2011). Gesture Recognition for American Sign Language with Polygon Approximation. *Proceedings of the IEEE International Conference on Technology for Education* (pp. 241-245).

Machacon, H., Shiga, S., & Fukino, K. (2012). Neural Network Application in Japanese Sign Language: Distinction of Similar Yubimoji Gestures. *Journal of Medical Engineering & Technology*, 36(3), 163-168.

Martinez, A. (2006). Three-Dimensional Shape and Motion Reconstruction for the Analysis of American Sign Language. *Proceedings of the Computer Vision and Pattern Recognition Workshop* (Vol. 1, pp. 146-146).

Mcguire, R. M., Hernandez-Rebollar, J., Starner, T., Henderson, V., Brashear, H., Ross, D., & Tech, G. (2004). Towards a One-Way American Sign Language Translator Engineering and Applied Science Brain and Cognitive Sciences. *Proceedings of the IEEE International Conference on Automatic Face and Gesture Recognition* (pp. 620-625).

Munib, Q., Habeeb, M., Takruri, B., & Al-Malik, H. (2007). American Sign Language (ASL) Recognition Based on Hough Transform and Neural Networks. *Expert Systems with Applications*, 32(1), 24-37.

Moni, M. A., & Ali, A. B. M. S. (2009). HMM Based Hand Gesture Recognition: A Review on Techniques and Approaches. *Proceedings of 2nd IEEE International Conference on Computer Science and Information Technology* (pp. 433–437).

Nguyen, T., & Ranganath, S. (2010). Recognizing Continuous Grammatical Marker Facial Gestures in Sign Language Video. *Proceedings of the Asian Conference on Computer Vision* (pp. 665-676).

Nguyen, T., & Ranganath, S. (2012). Facial Expressions in American Sign Language: Tracking and Recognition. *Pattern Recognition*, 45(5), 1877-1891.

Oz, C., & Leu, M. (2007). Linguistic Properties Based on American Sign Language Isolated Word Recognition with Artificial Neural Networks Using a Sensory Glove and Motion Tracker. *Neurocomputing*, 70(16-18), 2891-2901.

Oz, C., & Leu, M. (2011). American Sign Language Word Recognition with a Sensory Glove Using Artificial Neural Networks. *Engineering Applications of Artificial Intelligence*, 24(7), 1204-1213.

Parashar, A. (2003). *Representation and Interpretation of Manual and Non-manual Information for Automated American Sign Language Recognition*. University of South Florida.

Patel, I., & Rao, S. (2010). Technologies Automated Speech Recognition Approach to Finger Spelling. *Proceedings of the International Conference on Computing, Communication and Networking Technologies* (pp. 1-6).

Picone, J. (1990). Continuous speech recognition using hidden Markov models. *IEEE ASSP Magazine*, 7(3), 26–41.

Pugeault, N., & Bowden, R. (2011). Spelling It Out: Real-time ASL Fingerspelling Recognition. *Proceedings of the IEEE International Conference on Computer Vision Workshops* (pp. 1114-1119).

Rabiner, L. (1989). A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. *Proceedings of the IEEE*, 77(2), 257–286.

Rashid, O., Al-Hamadi, A., & Michaelis, B. (2009). A Framework for The Integration of Gesture and Posture Recognition Using HMM and SVM. *Proceedings of the International Conference on Intelligent Computing and Intelligent Systems* (Vol. 1, pp. 572-577).

Ricco, S., & Tomasi, C. (2009). Fingerspelling Recognition through Classification of Letter-to-Letter Transitions. *Proceedings of the Asian Conference on Computer Vision* (pp. 214-225).

Rodriguez, A., Weaver, J., & Pentland, A. (1998). Real-time American Sign Language Recognition Using Desk and Wearable Computer Based Video. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(12), 1371-1375.

Rybach, D. (2006). *Appearance-Based Features for Automatic Continuous Sign Language Recognition*. RWTH Aachen University

Sandler, W., & Lillo-Martin, D. (2001). Natural Sign Languages. In M. Aronoff & J. Rees-Miller (Eds.), *The Handbook of Linguistics* (pp. 533-562).

Sarkar, S. (2006). Detecting Coarticulation in Sign Language using Conditional Random Fields. *Proceedings of the International Conference on Pattern Recognition* (pp. 108-112).

Segouat, J., & Braffort, A. (2010). Toward Modeling Sign Language Coarticulation. *In S. Kopp & I. Wachsmuth (Eds.), Gesture in Embodied Communication and Human-Computer Interaction* (Vol. 5934, pp. 325-336). Berlin, Heigelberg: Springer Berlin Heigelberg.

Sethuraman, J., & Ranganath, S. (2007). Sign Language Phoneme Transcription with PCA-based Representation. *Proceedings of the International Conference on Information, Communications & Signal Processing* (pp. 1-5).

Sturman, D. J., & Zeltzer, D. (1994). A Survey of Glove-Based Input. *IEEE Computer Graphics and Applications*, 14(1), 30-39.

Singh, K. (2000). Skinning Characters using Surface-Oriented Free-Form Deformations. *Graphics Interface*, 35-42.

Silverman, H. F., & Morgan, D. P. (1990). The Application of Dynamic Programming to Connected Speech Recognition. *IEEE ASSP Magazine*, 7(3), 6–25.

Starner, T., & Pentland, A. (1995). Real-time American Sign Language Recognition from Video Using Hidden Markov Models. *Proceedings of the International Symposium on Computer Vision* (pp. 265-270).

Starner, T., Weaver, J., & Pentland, A. (1998). Real-time American Sign Language Recognition Using Desk and Wearable Computer Based Video. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(12), 1371-1375.

Szeliski, R. (2011). *Computer Vision : Algorithms and Applications*. London: Springer London.

Tanibata, N., Shimada, N., & Shirai, Y. (2002). Extraction of Hand Features for Recognition of Sign Language Words. *Proceedings of the International Conference on Vision Interface* (Vol. 1, pp. 391 - 398).

Taylor-DiLeva, K. (2010). *Once Upon A Sign: Using American Sign Language To Engage, Entertain, And Teach All Children* (p. 270). Santa Barbara, California, USA: Libraries Unlimited.

Thangali, A., Nash, J., Sclaroff, S., & Neidle, C. (2011). Exploiting Phonological Constraints for Handshape Inference in ASL Video. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 521-528).

Ullah, F. (2011). American Sign Language Recognition System for Hearing Impaired People Using Cartesian Genetic Programming. *Proceedings of the International Conference on Automation, Robotics and Applications* (pp. 96-99).

Vogler, C., & Metaxas, D. (1997). Adapting Hidden Markov Models for ASL Recognition by Using Three-Dimensional Computer Vision Methods. *Proceedings of IEEE International Conference on Systems, Man and Cybernetics* (Vol. 1, pp. 156-161).

Vogler, C., & Goldenstein, S. (2007). Facial Movement Analysis in ASL. Universal Access in the Information Society, 6(4), 363-374. Waldron, M. (1995). Isolated ASL Sign Recognition System for Deaf Persons. *IEEE Transactions on Rehabilitation Engineering*, 3(3), 261-271.

Wang, H., Stefan, A., & Athitsos, V. (2009). A Similarity Measure for Vision-Based Sign Recognition. *Proceedings of the International Conference on Universal Access in Human-Computer Interaction* (pp. 607-616).

Wang, X., Xia, M., Cai, H., Gao, Y., & Cattani, C. (2012). Hidden-Markov-Models-Based Dynamic Hand Gesture Recognition. *Mathematical Problems in Engineering*, 2012, 1–11.

Welch, L. R. (2003). Hidden Markov Models and the Baum-Welch Algorithm. *IEEE information Theory Society Newsletter*, 53(4), 10–13.

Wilson, E., & Anspach, G. (1993). Applying Neural Network Developments to Sign Language Translation. *Proceedings of the IEEE Neural Network for Signal Processing Workshop* (pp. 301-310).

Yang, M.-H., & Ahuja, N. (2002). Extraction of 2D Motion Trajectories and Its Application to Hand Gesture Recognition. *Pattern Analysis and Machine*, 24(8), 1061-1074.

Yang, R., & Sarkar, S. (2006). Detecting Coarticulation in Sign Language Using Conditional Random Fields. *Proceedings of the International Conference on Pattern Recognition* (pp. 108-112).

Yang, R., Sarkar, S., Member, S., & Loeding, B. (2010). Handling Movement Epenthesis and Hand Segmentation Ambiguities in Continuous Sign Language Recognition Using Nested Dynamic Programming. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(3), 462-477.

Yin, P., Starner, T., Hamilton, H., Essa, I., & Rehg, J. (2009). Learning the Basic Units in American Sign Language Using Discriminative Segmental Feature Selection. *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing* (pp. 4757-4760).

Yuan, Y. (2008). Step-sizes for the Gradient Method. *ASM/IS Studies in Advnaced Mathematics* (pp. 785–796).

Zafrulla, Z., Brashear, H., Hamilton, H., & Starner, T. (2010). A Novel Approach to American Sign Language (ASL) Phrase Verification Using Reversed Signing. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 48-55).

Zafrulla, Z., Brashear, H., Starner, T., Hamilton, H., & Presti, P. (2011). American Sign Language Recognition with the Kinect. *Proceedings of the International Conference on Multimodal Interfaces* (p. 279).

Zafrulla, Z., Brashear, H., Yin, P., Presti, P., Starner, T., & Hamilton, H. (2010). American Sign Language Phrase Verification in an Educational Game for Deaf Children. *Proceedings of the International Conference on Pattern Recognition* (pp. 3846-3849).

Zaki, M. M., & Shaheen, S. I. (2011). Sign Language Recognition Using a Combination of New Vision Based Features. *Pattern Recognition Letters*, 32(4), 572–577.

Zhang, Z., Alonzo, R., & Athitsos, V. (2011). Experiments with Computer Vision Methods for Hand Detection. *Proceedings of the 4th International Conference on PErvasive Technologies Related to Assistive Environments* (pp. 1–5).

Zhou, H., Lin, D. J., & Huang, T. S. (2004). Static Hand Gesture Recognition Based on Local Orientation Histogram Feature Distribution Model. *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops* (pp. 161–168).

APPENDIX A

A.1. Gamma Function

A Gamma function is defined as $\Gamma(x)=\int_0^\infty d\tau\,\tau^{x-1}e^{-\tau}$, which has a well know recursion $x!=\Gamma(x+1)=x\Gamma(x)=x(x-1)!$.

A.2. Dirichlet Distribution

The dirichlet distribution is as follows,

$$p(\theta\,|\,\alpha)=\frac{\Gamma(\sum_{s=1}^{m}\alpha_m)}{\Pi_{s=1}^{m}\alpha_s}\Pi_{s=1}^{m}\theta^{\alpha_s-1},$$

Where α_s is the s^{th} element of α, and $\Gamma(x)$ is the gamma function.

A.3. K-L Convergence

For the probability densities $p(x)$ and q(x) for $X\in D$ the KL-divergence is defined as follows:

$$KL(p\,||\,q)=\int_{x\in D}p(x)\log\frac{p(x)}{q(x)}.$$

A.4. Expectation of Logarithm Function of Dirichlet Distribution

$$E(\ln Dir(\lambda))=\int d\lambda Dir(\lambda\,|\,\lambda^*)\ln\lambda_j^*-\psi(\sum_{j=1}^{k}\lambda_j^*)$$

A.5. Digamma Function

The digamma function is defined as

$$\psi(x)=\frac{d}{dx}\ln\Gamma(x).$$

APPENDIX B

B.1. Maximum Likelihood

Maximum likelihood estimation (MLE) is a method of estimating the parameters of a statistical model. X_1, X_2, X_3,…, X_n have joint density denoted

$$f_\theta(x_1, x_2, ..., x_n) = f(x_1, x_2, ..., x_n \mid \theta).$$

Given observed values $X_1=x_1$, $X_2=x_2$,…,$X_n=x_n$, the likelihood of θ is the function

$$l(x \mid \theta) = f(x_1, x_2, ..., x_n \mid \theta),$$

which is considered as a function of θ.
In words, the likelihood function is the probability of observing the given observation as a function of θ. The MLE of θ is a value of θ that maximizes the likelihood, which means the value that makes the observed data the most probable.

$$MLE(\theta) = \max l(x \mid \theta).$$

Note that the solution to an optimization problem is invariant to a strictly monotone increasing transformation of the objective function, an MLE can be obtained as a solution to the following problem:

$$\max \log l(x \mid \theta) = \max L(x \mid \theta)$$

The EM algorithm is an efficient iterative procedure to compute the MLE. Convergence is assured since the algorithm is guaranteed to increase the likelihood at each iteration. However, depending upon the choice of the initial parameter values, the algorithm could prematurely stop and return a sub-optimal set of parameter values, which is called the local maxima problem. Unfortunately, there exists no general solution to the local maximum problem. Instead, a variety of techniques have been developed in an attempt to avoid the problem, though there is no guarantee of their effectiveness (Myung, 2003).

B.2. Mahalanobis Distance

In statistics, Mahalanobis distance is based on correlations between variables by which different patterns can be identified and analyzed. It gauges similarity of an unknown sample set to a known one. The Mahalanobis distance is defined as:

$$D^2 = (x-m)'C^{-1}(x-m),$$

where:

D^2 = Mahalanobis distance
x = Vector of data
m = Vector of mean values of independent variables
C^{-1}= Inverse Covariance matrix of independent variables

If the covariance matrix is the identity matrix, the Mahalanobis distance reduces to the Euclidean distance. If the covariance matrix is diagonal, then the resulting distance measure is called the normalized Euclidean distance.

B.3. Covariance

The first step in analyzing multivariate data is computing the mean vector and the variance-covariance matrix. Covariance is a measure of how much two random variables change together. The mean vector consists of the means of each variable. For covariance matrix, each element represents the relationship between two variables. If the matrix is diagonal, it means any variable is not related to any other ones, which indicates the variables are independent.

The covariance matrix of any sample matrix can be expressed in the following way:

$$Cov(x) = \frac{1}{n}\sum_{i=1}^{n}(x_i - \overline{x})(x_i - \overline{x})',$$

where x_i is the ith test sample, $\bar{x}$ is the mean vector of one class of training samples, and n is the number of test samples.